Model Based Pose Estimation from Uncertain Data

Thesis submitted for the degree "Doctor of Philosophy"

Yacov Hel-Or

Submitted to the Senate of the Hebrew University in Jerusalem (1993)

This work was carried out under the supervision of
Dr. Michael Werman and Professor Shmuel Peleg

Abstract

Model-based pose estimation is a process which determines the location and orientation of a given object relative to a specific viewer. The input supplied to the process consists of a description of the object, generally denoted the *model*, and a set of measurements of the object taken by the viewer. The study of pose estimation is important in many areas of computer vision, such as object recognition, object tracking, robot navigation, motion detection, etc. This thesis deals with similar problems of determining, from sensory data, the exact position and orientation of a $3D$ object represented by a model.

The difficulty in solving pose estimation problems is mainly due to the fact that the sensory data from which the pose should be determined is imprecise and noisy. A slight error in measurement may have a large effect on the precision of the solution or may not even allow any solution. The problem of imprecision is especially difficult when the measurements are filtered through a series of sensory and memory systems which add noise to the data. For instance, measurements obtained from an optical sensor is affected by noise due to the camera (chromatic distortions, imprecision in the camera model, etc.), sampling, digitization and imprecise processing of the raw data.

In this thesis we aim to overcome the problems of imprecision of the measured data and to produce a precise and stable solution while maximally exploiting the available data. We present a uniform formalism of computation in which measured data can be of any dimension (1D, 2D or 3D) and under any model of projection (orthographic or perspective). This uniform formalism allows simple and convenient fusion of various types of sensory data and also enables simple and efficient fusion of constraints, in the case of constrained objects.

The thesis deals with two important cases of the problem of pose estimation:
The first part deals with determining the pose of a rigid $3D$ object where the sensory data is obtained from various kinds of sensors. We deal with models consisting of a set of feature points, such as maximum curvature, segment endpoints or corners. The measurements taken on these points are noisy and can be of various dimensions: $1D$, $2D$, or $3D$. In the case of $2D$ or $1D$ measurements, the projection model can be either orthographic or projective. This part of the thesis presents a method for fusing all these types of measurements in order to obtain a precise and stable estimate of the object's pose. Unifying the different types of measurements is done by associating an uncertainty matrix with each measured feature. The uncertainty depends both on the measurement noise and on the type of measurement. For example, a $2D$ measurement is a projection (perspective or orthographic) onto a $2D$ plane (as in a picture) and we regard it as a measurement in $3D$ with infinite uncertainty in the direction of the projection. Therefore,

the dimensionality of the measurements is encoded in the covariance matrix. With this paradigm we obtain a uniform mathematical formulation of the problem and can fuse various kinds of measurements to obtain a better and stable solution. The measurement fusion is performed using the Kalman Filter.

The second part of the thesis extends the solution described for estimating pose of rigid objects, to deal with articulated and constrained objects. A constrained object is an object composed of a set of rigid components having spatial constraints between them. An articulated object is a special case of a constrained object, where the components are connected at joints which allow certain degrees of freedom. These joints impose constraints on the component locations, since the pose of one component influences the pose of the other. We do not limit the constraints to any single type and allow any type of constraint including inequality constraints. In the method we suggest, both measurements and constraints are treated similarly while varying only their associated uncertainty. The constraints are considered as perfect "measurements" with zero uncertainty whereas the measurements themselves (the actual measurements) have uncertainty greater than zero. In other words the actual measurements are considered *soft* constraints whereas the constraints are considered *strong*.

The solution we suggest to solve the pose estimation problem for rigid and constrained objects is advantageous due to the following properties:

- The generality of the method -
  The main concept in our approach is the presentation of a general and uniform formalism to compute the pose of an object where the measurements are of any dimensionality and under any projection model (orthographic or perspective). This formalism allows not only a uniform approach to measurements of different types, but also allows a similar approach to various types of constraints (when dealing with constrained objects).

- Integration of information -
  The uniform representation and uniform processing of all types of measurements, enables simple and efficient fusion of information of different types and from different sources inorder to determine the pose of an object. The fusion of different types of data is inherent in the algorithm and allows the system to produce a precise and stable solution while optimally exhausting the available input data.

- Incremental process -
  In many of the existing methods the pose of an object is evaluated using a batch process, i.e. all the measurements are considered simultaneously and are used in a single vector equation. In such a process, the correspondences between measured points and model points must be assumed as given, so that all the equations can

be generated before calculating the solution. The assumption that the correspondences are known, is often not acceptable, specifically in context of recognition. The approach we adopt is in accord with the paradigm suggested in Faugeras and Hebert [22] where the pose estimation and the correspondence problems are solved simultaneously in an incremental manner. At each stage, the pose of the object is estimated from the previously matched points. This estimate assists in rejecting irrelevant correspondences in the current stage.

- Incorporating constraints into the solution -
  When estimating the pose of a constrained object, the method we suggest allows simple and efficient fusion of constraints into the solution. Thus, information obtained on the pose of any single component of the object, is propagated to all other components through the mutual constraints. In this manner, the estimated solution takes into account all the existing measurements and all the defined constraints.

- Using the Kalman Filter -
  The suggested solution uses Kalman Filter (K.F.) tools and so includes the advantages associated with the K.F. such as explicitly dealing with the measurement uncertainty, simple updating of the solution given additional measurements, and easy parallelization [33].

Contents

Chapter 1

Introduction

Model-based pose estimation is a process which determines the location and orientation of a given object relative to a specific viewer. The input supplied to the process consists of a description of the object, generally denoted the *model*, and a set of measurements of the object taken by the viewer. For example, consider a robot whose task it is to grasp and raise a wooden block from a table. In order to accomplish its task, the robot must know the exact position of the block relative to its arm. Thus the robot optically senses its surrounding including the table and the wooden block (using a CCD camera, an infrared sensor, a range finder, etc) and tries to extract the required pose information in order to grasp the object.

The study of pose estimation is important in many other areas of computer vision, such as object recognition, object tracking, robot navigation, motion detection, etc. This thesis deals with similar problems of determining, from sensory data, the exact position and orientation of a $3D$ object represented by a model.

This thesis deals with the following two cases:

- Finding the pose of a rigid object.
- Finding the pose of a constrained object.

A constrained object is an object composed of a set of rigid components having spatial constraints between them. An articulated object is a special case of a constrained object,

where the components are connected at joints which allow certain degrees of freedom. These joints impose constraints on the component locations, since the pose of one component influences the pose of the other. We do not limit the constraints to any single type and allow any type of constraint including inequality constraints.

The difficulty in solving pose estimation problems is mainly due to the fact that the sensory data from which the pose should be determined is imprecise and noisy. A slight error in measurement may have a large effect on the precision of the solution or may not even allow any solution. The problem of imprecision is especially difficult when the measurements are filtered through a series of sensory and memory systems which add noise to the data. For instance, measurements obtained from an optical sensor is affected by noise due to the camera (chromatic distortions, imprecision in the camera model, etc.), sampling, digitization and imprecise processing of the raw data. In this thesis we aim to overcome the problems of imprecision of the measured data and to produce a precise and stable solution while maximally exploiting the available data. We present a uniform formalism of computation in which measured data can be of any dimension (1D, 2D or 3D) and under any model of projection (orthographic or perspective). This uniform formalism allows simple and convenient fusion of various types of sensory data and also enables simple and efficient fusion of constraints, in the case of constrained objects.

1.1 The Model

The object, whose pose is to be determined, is represented by a *model* which is a set of features and attributes describing the object. In the articulated case the object is composed of a set of rigid components connected at joints which allow certain degrees of freedom. These joints can be, for example, *prismatic joints* which allow relative translation between components, or *revolute joints* which allow relative rotation of the components about a point. Each object joint enforces a constraint on the position of the components. Therefore in this case, the model includes, in addition to the representation of each component, a set of constraints which describe the mutual relationships between the components. In fact we extend the definition of the problem to models that include additional general constraints such as co-linearity or co-planarity of the model components, angle relationships, etc. The constraints may also include inequality constraints

such as limited range of distances between points or limited range of angles. We call this kind of models *constrained models*.

Finding the pose of an object in space is in essence determining the possible transformation that transforms the model to the pose in space from which it is observed by the sensor. In the case where the object is rigid this transformation is composed of a rotation followed by a translation. In the constrained case a rigid transformation should be calculated for each one of the components.

The methods of determining pose are dependent on the manner by which the object is represented. A wide variety of object representations have been suggested in the literature. These can be generally divided into two main categories [10, 41]:

- Shape representation.
- Feature representation.

A *shape representation* represents the object by a description of the general shape or structure of the object. This representation can be divided into three groups according to the dimensionality of the descriptors (model elements) [10]:

1. volumetric representation - the object is described by a set of volumetric primitives such as constructive solid geometry [62], octree representation [56] and generalized cylinders [13].
2. surface representation - the object is described by its external surfaces either using surface patches [60] or using a parametric description [59].
3. representation by curves- the object is described by a set of representative curves. For example, using skeletons [77] or wire frames [8].

Shape representation of an object is commonly used in computer graphics since the appearance of the object from any given view-point, can be easily predicted. However, for the purpose of pose estimation, a more compact representation of the object is sufficient. In this context, shape representation is superfluous and inefficient.

Furthermore, the process of comparing such a shape representation model with sensory data is indirect and relatively complicated to implement.

A *feature representation* of an object is better suited to the task of pose determination due to its compactness. Such a representation consists of a set of features which describe different entities of the object. The types of features chosen to represent an object can be divided into two classes:

1. local features - every entity is locally evaluated and represents a small part of the object. Examples for local features are: corners, maximum curvature points, inflection points, angles, segments, etc (e.g. [16]).

2. global features - every entity is evaluated from the entire object, for example, the center of mass, the moments of inertia, Fourier descriptors, etc. (e.g. [39, 63]).

Although global features are relatively robust to measurement noise, determining the pose of an object based on global features is problematic since such a representation is sensitive to partial occlusion. Additionally, global features are generally not invariant under projections [80] so estimating the pose of a 3D object from 2D sensory data is not straight forward.

Most studies on pose estimation, including this thesis, focus on local feature representations. In addition to being a compact representation, local features are not sensitive to partial occlusions. Furthermore, the extraction of local features from the sensory data is a local and relatively simple process which can easily be parallelized. However, since the local features are extracted from a limited part of the input, they are sensitive to noise. Another disadvantage is the need to find an interpretation for the measurements, i.e. the need to correspond every measured feature with a feature of the model. This problem is also known as the "correspondence problem". In order to reduce the effect of noise on the pose estimation, a large number of local features should be used. Additional reduction can be obtained by integration of measurements from different types of sensors and by carefully processing the sensory data to take into account the measurement noise. The main theme of this thesis is to exhaust the maximum amount of information about the pose of a measured object where its representation is given by local features.

The correspondence problem and the ways to overcome it will be dealt with later in this section.

1.2 The Sensory Data and its Uncertainty

Sensory data is obtained from measurements taken on the object. This is the source of information from which the pose of the object should be determined. In this thesis we do not limit ourselves to any specific type of measurement nor to any specific projection model and allow the data to be from different types of sensors. Since the measurements are noisy, we will require that each measurement be associated with its uncertainty estimate which defines a measure of the "inexactness" of that measurement. If we consider the measurement as a statistical process which samples values from a specific distribution, the uncertainty of the measurement is then defined as the variance of the distribution or, in the case of a multi-dimensional measurement, as the covariance matrix. Several studies deal with characterizing and modeling the measurement noise. However, in general it is difficult to give an exact description of the noise distribution because of the large number of noise sources and because of the complexity of the noise model of each source. Measurements obtained from optical imaging, for example, are contaminated by noise originating from chromatic aberrations, inexact camera model, digitization, incorrect feature extraction etc. We do not deal with noise modeling, but we assume a given Gaussian process with known parameters. This assumption is based on the central limit theorem [19], which translates to the fact that as the number of noise sources increases, the distribution of the total noise becomes more similar to a Gaussian distribution. This assumption is not always consistent with reality, since quite often there are gross-errors which deviate from the average and which cause the Gaussian distribution to contain outliers. In Section 3.6 we further discuss the justification in describing the noise distribution as a Gaussian distribution.

1.3 The Correspondence Problem

As previously noted, we solve the pose determination problem using local features for the description of the object. This requires a matching between every measured feature and

a corresponding model-feature. An important aspect of the correspondence problem is the dimensionality of the model features vs. the dimensionality of the measured features. The common cases are the following [30]:

- $3D$ to $3D$ correspondence: both model and measurements give the $3D$ location of features (measurements from range data, stereo, etc.).
- $2D$ to $3D$ correspondence: the model is $3D$ while the available measurements supply projected information in $2D$, such as locations in an image plane.

The 2D to 2D correspondence problem, where both model and measurements are two dimensional, can be included in the 2D to 3D correspondence problem, since a 2D model can be viewed as a planar 3D model.

Regardless of the dimensionality of the model and measured features, the matching process is time consuming since the number of possible matches is exponential in the number of features. Thus, an exhausted search for the correct match is impractical. Several methods have been suggested to reduce the cost of this search. Basri [10] classifies them into three groups:

- Minimal alignment methods.
- pruning the search in the correspondence space.
- pruning the search in the transformation space.

The minimal alignment methods (M.A) are based on the fact that rigid transformations can be determined by a small number of matches. For example, in the case of $3D$ objects where the $3D$ locations of feature points are measured, three model points and their corresponding measurements determine the rigid transformation [42]. These methods find the transformation in two stages:
In the first stage, a minimal number of matches are chosen and a corresponding transformation is determined. In the second stage, a $verification$ is performed to ensure that the remaining model features are consistent with the measurements. If the verification fails, i.e. the transformed model does not match the measurements, then the whole process is repeated. Using this method, the correspondence problem can be calculated in

polynomial time in the number of features rather than exponential. Examples of system using M.A. method can be found in [42, 10, 69].

The main problem in using M.A. is the inaccuracy of the computed $3D$ transformation due to the small number of matches used in the estimation process. However, this method can be used to obtain an initial estimate of the transformation, which can be used as a guide to find additional consistent matches. After additional matches are accumulated, a better pose estimate can be recalculated. This paradigm is similar to the method suggested by Faugeras at el. [22, 20, 6] which will be described later.

The methods which try to prune the search in the transformation space, choose, from among the set of all possible transformations, that transformation which is consistent with the largest number of matches. The generalized Hough transform [9] is such a search process, which is performed as follows: The transformations are represented in the space of their parameters, and the possible range of values of the parameters is quantized into a finite number of bins which serve as accumulators. For every pairing between a model feature and a measured feature, the set of all possible transformations that align these two features is evaluated, and the corresponding bins in the quantized space are increased by one vote. The transformation which received the greatest number of votes determines the best correspondence between the model features and the measured features.

There are two main disadvantages in using this method: Since the transformation space is of high dimensionality (6 dimensions for rigid transformation in $3D$), the table of accumulators is very large and the voting is time consuming. However, quantizing into larger bins will result in an imprecise transformation. An additional disadvantage to this method is that the optimal transformation may be missed due to quantization, specifically when measurement noise causes the optimal transformation to be distributed between several bins.

The methods which reduce the space of possible matches regard the correspondence problem as a search problem in a graph. This graph defines a pairing between the model features and the measured features. The basic scheme behind these methods is to prune parts of the graph which represent impossible pairings. Faugeras [22, 20, 6] and Grimson [29] follow a similar method. They represent all possible matches in an *interpretation tree* (I.T.). in which every level corresponds to a measured feature, and every node

in that level denotes a match between the measured feature and some model feature. Every path in the I.T., from root to leaf, defines a full interpretation, i.e, defines for each measurement the corresponding model feature. Both, Grimson and Faugeras search for a consistent path in the I.T., using depth first search. Grimson restricts the search by defining local constraints in the model, such as limiting the range of distances and angles between model features. Any branch of the I.T. which contradicts the local constraints is pruned. Faugeras evaluates for every partial path the transformation which is consistent with the matches in the path. This transformation is applied on the model features and these are used to further restrict the remaining matches. In this paradigm the pose of the object is solved simultaneously with the correspondence problem.

A detailed discussion of the correspondence problem is not in the scope of this thesis, however, in this work, we do consider the possibility of obtaining a correct interpretation of the measured features. Thus, the methods suggested in this thesis for solving the pose estimation problem are incremental and enable a sequential process of matching to be combined with the pose determination process. This approach is in accord with the paradigm suggested by Faugeras.

1.4 Existing Approaches to Pose estimation

The problem of determining the pose of an object from measurements is dealt with intensively in the area of photogrametry and computer vision. Most studies deal with pose estimation of rigid objects and a large variety of solutions have been discussed and can be found in the literature (for reviews see [64, 30, 76]). Fewer studies deal with pose estimation of constrained and articulated objects (e.g. [12, 27, 26, 52, 51, 57, 66]). The solutions for pose estimation, whether for rigid or constrained objects, differ in the following main aspects [64]: (i) the features used in describing the object and model. (ii) the sensor model and the type of measurements available as inputs. (iii) the parameters of the transformation, to be estimated. (iv) the mathematical and computational methods used in the solution. (v) the methods of introducing constraints into the system (when dealing with constrained objects).

The features used to describe objects and models are either global (such as Fourier

descriptors [80], moments of the object [70], etc.) or local features representing small regions or parts of the model or object (for example points, lines, planes etc.).

The sensor model and the dimensionality of the sensed measurements vary among the different pose estimation methods. Dimensionality of the measurements is usually assumed to be either 3D or 2D. In the case of 2D measurements, the sensor model must include a projection model which describes the transformation of a point from 3D coordinates to 2D coordinates. The two most common projection models are the *perspective projection* model and the *orthographic projection* (weak perspective) model [36]. Another important difference between existing pose estimation methods concerns the treatment of the uncertainty associated with a given measurement (or associated with a sensor model). Several methods do not deal with these uncertainties at all (e.g. [29, 24, 38]), which is analogous to associating a uniform and isotropic uncertainty to all the measurements. Other methods take into account the uncertainties either as a scalar value associated with each measurement (representing an isotropic uncertainty for each measurement) [67, 30] or by spatial uncertainty usually represented by a covariance matrix (which can represent both isotropic and non isotropic uncertainties) [5, 7].

The mathematical and computational methods used to solve the pose estimation problem strongly depend on the representation chosen for the transformation to be estimated. The computational methods are basically of two types: incremental methods and batch methods. In the incremental methods, the solution is initially estimated according to some subset of the input measurements and then continually updated using additional measurements [22, 20]. The batch methods evaluate the solution in a single computation by either considering all available measurements and finding the optimal solution under some criterion [49, 3, 67, 37] or by calculating the solution using a minimal number of measurements and then verifying that most other measurements are consistent with the calculated solution [42, 24, 69].

Finally, when dealing with pose estimation of constrained and articulated objects, the existing solutions vary in the techniques of introducing the constraints into the system. In several solutions [26, 12, 66], the constraints of the model are not considered during the estimation process though they may be verified at the end of the process. In this case the pose of each component of the object is estimated using only its measurements and no

mutual information transfers between components. In other methods, the constraints of the model are eliminated by reparameterizing the model and reducing the number of the estimated parameters to be the number of degrees of freedom of the system [51, 52, 57]. With this technique no constraints remain in the new parameterization.

For a more detailed discussion on existing pose estimation approaches see Chapter 2.

1.5 Overview of this Thesis and its Contribution

This dissertation is divided into two main parts:
The first part (Chapter 4) deals with determining the pose of a rigid $3D$ object where the sensory data is obtained from various kinds of sensors. We deal with models consisting of a set of feature points, such as maximum curvature, segment endpoints or corners. The measurements taken on these points are noisy and can be of various dimensions: $1D$, $2D$, or $3D$. In the case of $2D$ or $1D$ the projection model can be either orthographic or projective. This part of the thesis deals with a method for fusing all these types of measurements in order to obtain a precise and stable estimate of the object's pose. Unifying the different types of measurements is done by associating an uncertainty matrix with each measured feature. Uncertainty depends both on the measurement noise and on the type of measurement. A $2D$ measurement is a projection (perspective or orthographic) onto a $2D$ plane (as in a picture) and we regard it as a measurement in $3D$ with infinite uncertainty in the direction of the projection. A $1D$ measurement provides only the distance from the observer to a point, and we regard it as a $3D$ vector with infinite uncertainty in its direction and finite variance in its length. Therefore, the dimensionality of the measurements is encoded in the covariance matrix. With this paradigm we obtain a uniform mathematical formulation of the problem and can fuse various kinds of measurements to obtain a better and stable solution. The measurements fusion is performed using the Kalman Filter [43, 55].

The second part of the thesis (Chapter 5) extends the solution described for estimating pose of rigid objects, to deal with articulated and constrained objects. The constraints between the object components can be of general types and may also include inequality constraints. In the method we suggest, both measurements and constraints are treated

similarly while varying only their associated uncertainty. The constraints are considered as perfect "measurements" with zero uncertainty whereas the measurements themselves (the actual measurements) have uncertainty greater than zero. In other words the actual measurements are considered *soft* constraints whereas the constraints are considered *strong*.

The solution we suggest to solve the pose estimation problem for rigid and constrained objects is advantageous due to the following properties:

- The generality of the method -
  The main concept in our approach is the presentation of a general and uniform formalism to compute the pose of an object where the measurements are of any dimensionality and under any projection model (orthographic or perspective). The uniformity in dealing with all types of measurement is possible due to the uniformity in considering all measurements as 3D measurements with a relevant 3X3 covariance matrix. This formalism allows not only a uniform approach to measurements of different types, but also allows a similar approach to various types of constraints (when dealing with constrained objects). The constraints are also considered as measurements where the associated uncertainty is zero.

- Integration of information -
  The uniform representation and uniform processing of all types of measurements, enables simple and efficient fusion of information of different types and from different sources inorder to determine the pose of an object. The fusion of different types of data is inherent in the algorithm and allows the system to produce a precise and stable solution while optimally exhausting the available input data. Additionally, when dealing with constrained objects, multiple constraints of different types can be simply included in the model.

- Spatial uncertainty of the measurements -
  In many of the existing methods of pose estimation (see Chapter 2), the treatment of uncertainties of the measurements is performed in a general manner by assigning a scalar value to each measurement representing a weight proportional to the reliability of the measurement. For example, a 3D measured point represented by three

parameters $\mathbf{p}'_i = (x'_i, y'_i, z'_i)^t$ and by a covariance matrix Λ, is assigned a weight w_i representing the reliability of the measurement. In most cases, this weight is inversely proportional to trace(Λ). When generalizing uncertainties in this manner, not all information available to the system is exploited. For example, a measurement of a point at the end of a segment has small uncertainty in the direction perpendicular to the segment and large uncertainty in the direction parallel to the segment. A scalar weight assigned to this measurement will have a small value even though part of the measurement is of high certainty. The method we suggest, deals explicitly with the spatial uncertainty of each one of the measurements and thus maximally exploits the information available from the measurements.

- Uncertainty of the solution -
  Most of the existing methods supply as the solution, only the values of the parameters that were estimated without supplying a measure of the quality of the solution. The method suggested in this thesis supplies, in addition to the estimated pose, an estimate of the quality or reliability of the solution. This measure of quality which is influenced by the uncertainties of the measurements can be represented as a covariance matrix of the estimated parameters. Evaluating the quality of the solution is important especially in order to sequentially fuse information from several measurements. In this case the quality of the solution continuously increases as additional measurements are fused (or the uncertainties decrease). The sequential fusion of measurements is important when no correspondences are given between measured features and model features as described in Section 1.3.

- Incremental process -
  In many of the existing methods the pose of an object is evaluated using a batch process, i.e. all the measurements are considered simultaneously and are used in a single vector equation. In such a process, the correspondences between measured points and model points must be assumed as given, so that all the equations can be generated before calculating the solution. The assumption that the correspondences are known, is often not acceptable, specifically in context of recognition. The approach we adopt is in accord with the paradigm suggested in Faugeras and Hebert [22] where the pose estimation and the correspondence problems are solved

simultaneously in an incremental manner. At each stage, the pose of the object is estimated from the previously matched points. This estimate assists in rejecting irrelevant correspondences in the current stage. Such a process requires an incremental algorithm for pose estimation since solving the problem in batch mode at every stage, is highly time consuming. The estimation obtained from non linear equations using an incremental process is less precise than using a batch process (see the Cramer Rao bound in [79]). However, the advantage in the ability to perform the matching during the estimation process, turns the scales.

- Models with general types of constraints -
  The existing pose estimation methods that deal with constrained objects are restricted to deal with articulated models. They deal with constraints that are due to prismatic or revolute joints between the model components. In our method we are not limited to any type of constraints and can deal with all types of constraints including co-linearity, co-planarity, constant distance, constant angle, etc. Additionally, we deal with inequality constraints such as limited range of distances between points or limited range of angles.

- Incorporating constraints into the solution -
  When estimating the pose of a constrained object, the method we suggest allows simple and efficient fusion of constraints into the solution. Thus, information obtained on the pose of any single component of the object, is propagated to all other components through the mutual constraints. In this manner, the estimated solution takes into account all the existing measurements and all the defined constraints. Each constraint is simply treated since it is locally defined (i.e. each constraint defines a relation between neighboring components, independent of all other components). However, its influence is global (influencing also those components which are not directly connected). Here too, the constraints can be added incrementally, allowing an efficient matching strategy.

- Using the Kalman Filter -
  The suggested solution uses Kalman Filter (K.F.) tools and so includes the advantages associated with the K.F. such as explicitly dealing with the measurement

uncertainty, simple updating of the solution given additional measurements, and easy parallelization [33].

1.6 The Organization of this Thesis

In the following section (Chapter 2) we give a more detailed discussion of existing work on pose estimation of both rigid and constrained objects. Since, most of the problems discussed in this thesis can be expressed as parameter estimation problems, we briefly review, in Chapter 3, the topic of parameter estimation, and describe the tools used in this thesis. These include several versions of the Kalman filtering which were developed for control of dynamic systems but are also used in this work.

In Chapter 4 we describe the general framework of our method of pose estimation as applied to rigid objects. In Chapter 5, we develop the pose estimation method to deal with constrained objects.

Chapter 2

Overview of Related Work

The problem of determining the pose of an object from measurements is dealt with intensively in the area of photogrametry and computer vision. Most studies deal with pose estimation of rigid objects and fewer studies deal with pose estimation of constrained and articulated objects. The solutions for pose estimation, whether for rigid or constrained objects, differ in the following main aspects [64]:

1. The features used in describing the model.
2. The sensor model and the type of measurements available as inputs.
3. The parameters of the transformation, to be estimated.
4. The mathematical and computational methods used in the solution.
5. the methods of introducing constraints into the system (when dealing with constrained objects).

2.1 Features of the Model

Several studies deal with finding the pose of an object, where the given model is described by global features, such as Fourier descriptors [80] or moments of the object [70]. Since, in our work, we chose to represent an object by local features, we focus in this review on previous studies that deal with models described by local features.

In most studies that deal with local features, the models are described by one or a combination of the following features:

1. Points (for example in [38, 29, 67, 76, 3, 24, 81, 7, 30, 35]).
2. Lines or line segments (as in [49, 22, 20, 7, 15]).
3. Planes (for example in [22, 21, 7, 15]).
4. Quadrics (as in [22, 15]).

The most common model description to be found in the literature is that given by point features. When dealing with point features in 3D, at least 3 model points and three corresponding point measurements are required in order to determine the pose of a 3D object [42]. When the measurements are taken following a projection (2D measurements), at least six model points and their corresponding measured points are required in order to determine the pose [24, 50, 24, 25]. However, in order to reduce the effects of measurement noise on the precision of the solution, several studies, including our work, use a larger number of points (e.g. [3, 15, 22, 30, 67, 38]). The advantage in using point features in the pose estimation solution, is the relative ease of extracting these features from images. However, in many cases, finding feature points on objects is not simple, specifically when objects consist of smooth surfaces.

Several studies on pose estimation represent the object model using line features (for example [49, 22, 7, 15]). At least three 3D lines or eight 2D lines are required as measurements in order to determine uniquely the pose of an object [42, 49]. The main advantage in using line features (when the measurements are in $3D$) is that the estimation of the translation and the estimation of the rotation can be easily decoupled. This is due to the fact that the $3D$ orientation of a measured line is independent of translation. The drawback in using line features is that more lines are required to uniquely determine the pose of an object.

Including surfaces and quadrics in the pose estimation process (e.g. [22, 21, 7, 15]), greatly enriches the description of the object and enables description of objects having smooth surfaces. However, they are applicable only with 3D measurements.

Since the work presented in this presentation uses feature points, we concentrate the review on studies dealing with pose estimation from feature points.

2.2 Representation of the Transformation

The methods and algorithms for determining the transformation of an object relative to the sensor, is tightly linked with the transformation representation. The transformation consists of two components: a component that describes the rotation of the object coordinate frame relative to the sensor coordinates and a component that describes the relative translation between the two coordinate frames. The translatory component is usually described by a three dimensional vector t which gives the translation in the x,y and z axis directions. However, the rotational component can be described in several methods, which influence the choice of computational method to be used for the estimation process [64]. In the following we review some representations of rotations which are common in the literature.

Orthogonal Matrix

The most common method of representing rotation in 3D is using a 3x3 matrix R which represents a linear transformation in affine space. The matrix R is composed of 9 parameters, however these are not linearly independent; the characteristics of a rotation matrix which distinguishes it from any other linear transformation are that R is orthonormal i.e.

$$RR^t = I$$

(I is the identity matrix) and that $det(R) = 1$. The orthonormality restriction, imposes 6 nonlinear constraints on the 9 parameters thus there are 3 degrees of freedom in a rotation matrix. If $\mathbf{p}'$ is the coordinates of a point p which has undergone rotation then $\mathbf{p}' = R\mathbf{p}$.

The matrix representation which is commonly used, is attractive mainly due to the simplicity in dealing with 3x3 matrices. The main disadvantage in using this representation is the large number of parameters to be estimated and the inconvenience in dealing

with six nonlinear constraints. Some examples of studies using this representation can be found in [38, 3, 81, 15, 35].

Euler Angles

Any 3D rotation can be described as a sequence of rotations by angles (θ, ϕ, ψ) about the x,y and z axes respectively [64]. These three angles, called the Euler Angles, form three free parameters that describe any rotation transformation. Any 3D rotation is uniquely defined by these three angles, except for the case where $\theta = \pm\pi/2$ when ϕ and ψ cannot be determined uniquely [64]. It can be shown that any minimal parameterization of the rotation group must contain points of singularity [37]. Some examples of studies that represent rotations using Euler Angles can be found in [49, 30].

Quaternions

A rotation can be described by a rotation angle θ and a unit vector $\hat{\mathbf{n}}$ representing the rotation axis. The unit quaternion is defined as four parameters:

$$\tilde{\mathbf{q}} \;=\; (q_0, q_x, q_y, q_z)^t \;=\; (\cos\tfrac{\theta}{2}, \sin\tfrac{\theta}{2}\hat{\mathbf{n}})$$

These four parameters define a rotation operator in $3D$ according to arithmetic rules specific for quaternions (see Appendix B). Representing a rotation by quaternions requires the estimation of four parameters under a single constraint $\|\tilde{\mathbf{q}}\|^2 = 1$. This parameterization forms a two to one mapping over the rotational group, since a rotation of angle θ about the axis $\hat{\mathbf{n}}$ is equivalent to a rotation of angle $-\theta$ about the axis $-\hat{\mathbf{n}}$. Although the mapping is two to one, this raises no problem since the mapping is a local homeomorphism [64].

Representing rotations by quaternions is convenient, since in addition to there being only a single constraint, the rotation is linear in $\tilde{\mathbf{q}}$. Several studies represent rotations by quaternions, for example [22, 67, 20, 21, 37].

Exponential of Skew Matrices

Another method of representing a rotation R in 3D is by an exponent of a skew-symmetric matrix H:

$$R = e^H$$

where the exponential of a matrix is defined by the series

$$e^H = I + \frac{H}{1} + \frac{H^2}{2} + \dots$$

It can be shown that if the skew-symmetric matrix H is given as:

$$H = \begin{pmatrix} 0 & -c & b \\ c & 0 & -a \\ -b & a & 0 \end{pmatrix}$$

then e^H represents a rotation by $\theta = (a^2 + b^2 + c^2)^{1/2}$ about the axis $\mathbf{r} = (a, b, c)^t$ [73]. Similar to the Euler Angles representation, the representation by an exponential of a skew-symmetric matrix is a minimal representation (having three parameters). However, it has points of singularity and the mapping onto the rotational group is a two to one mapping [64, 7]. Examples of studies using this representation can be found in [7, 6].

2.3 Dimensionality of the Measurements

The methods of pose estimation of a 3D object are generally divided into two groups according to the dimensionality of the input measurements [30]:

1. Methods using 3D measurements.
2. Methods using 2D measurements (taken following a projection).

In the following, we briefly review the principle methods in these two groups where the object is rigid. A good review on the first group can be found in [64] and on the second group in [81]. Following we review the existing methods for the constrained and articulated case.

2.4 Rigid Case: Methods for Pose Estimation from 3D Measurements

We first review several closed form solutions and then we review some iterative methods.

2.4.1 Closed Form Solutions

Direct Linear Methods

The most convenient and straight forward method of evaluating the rotation R and the translation $\mathbf{t}$, is by simply disregarding the 6 constraints existing on R and estimating the 9 parameters of matrix R and the 3 parameters of $\mathbf{t}$ (see for example [14]). Every measured point $\mathbf{p}'$ supplies a set of 3 linear equations:

$$\mathbf{p}' = R\mathbf{p} + \mathbf{t}$$

where $\mathbf{p}$ is the $3D$ model point corresponding to $\mathbf{p}'$. Thus the 12 parameters can be estimated from at least 4 linearly independent measurements [14].

Although simple, this method is not practical since it produces a true rigid transformation only in the ideal cases when the measurements are free of noise [64]. In the general case, when measurements are noisy, this method does not give the correct solution. Furthermore, not all information in the system is being exploited (the fact that the transformation is rigid) thus the solution obtained is not expected to be optimal.

Methods Based on Translation Invariants

In order to find rigid transformation more complex methods must be used. Several methods approach this problem by first finding the rotation R and then evaluating the translation $\mathbf{t}$ using the estimated rotation. One method of finding the rotation independently of the translation is by using directional measurements rather than locational measurements, since the former are invariant to translation. Directional measurements can be obtained when the model features that are used are line primitives or planar primitives. In these cases the normal of these primitives serve as directional measurements [20, 22, 29].

Another possibility is to consider the vector difference, between two measured points, as a directional feature: $\mathbf{v}'_{ij} = \mathbf{p}'_i - \mathbf{p}'_j$. In this case $\mathbf{v}'_{ij} = R\mathbf{v}_{ij}$ where $\mathbf{v}_{ij}$ is the vector difference between the two corresponding model points [67, 14]. A third possibility is to describe the set of model points and the set of measured points relative to their center of mass [37, 3, 81, 38]. This representation is also translation invariant.

After finding an estimate $\hat{R}$ for the rotational component of the transformation, the translation $\mathbf{t}$ can be easily estimated from a set of linear equations of the from: $\hat{R}\mathbf{p}_i - \mathbf{p}'_i = \mathbf{t}$. The disadvantage in using these methods is that any error in estimating the rotation will induce an error in the translation estimate as well. Furthermore, the estimate obtained using a representation of points relative to their center of mass, is sensitive to partial occlusion.

Least Squares solution using Quaternions

Horn [37] represents model features and measured features relative to their center of mass and represents rotations using quaternions. Following this method, every measurement supplies a quaternion equation: $\tilde{\mathbf{v}}_i = \tilde{\mathbf{q}}\tilde{\mathbf{v}}'_i\tilde{\mathbf{q}}^*$, where $\tilde{\mathbf{q}}^*$ is the conjugate quaternion of $\tilde{\mathbf{q}}$ and $\tilde{\mathbf{v}}_i$, $\tilde{\mathbf{v}}'_i$ are the coordinates of the model point and the corresponding measured point relative to their center of mass and represented in quaternion form (see Appendix B). Horn shows that finding the optimal quaternion under the L.S. criterion (least squares) and under the constraint $\|\tilde{\mathbf{q}}\|^2 = 1$, is equivalent to finding the eigen vector corresponding to the smallest eigen value of a 4x4 matrix.

Grimson and Lozano-Perez [29] and Faugeras and Hebert [22] follow a similar solution for estimating the rotation, where the measure features are planes and the point coordinates are replaced by normal vectors of the planes.

Adapted Spherical Projection Method

Blosten and Huang [14] represent the rotational component of the transformation by a rotation axis $\mathbf{r}$ and an angle θ. Every two difference vectors $\mathbf{v}_{ij}$ and $\mathbf{v}_{kl}$ (where $\mathbf{v}_{ij} = \mathbf{p}_i - \mathbf{p}_j$) and their correspondences supply an equation over the rotation axis:

$$\mathbf{r} = (\mathbf{v}'_{ij} - \mathbf{v}_{ij}) \times (\mathbf{v}'_{kl} - \mathbf{v}_{kl})$$

Thus the axis $\mathbf{r}$ can be estimated from 3 measured points and their 3 corresponding model points. Following the estimation of $\mathbf{r}$, the angle θ can be estimated from the Rodriguez formula (see [44]). The drawback of this method is that $\mathbf{r}$ is very sensitive to noise in the measurements since it is estimated from only three measured points. Grimson and Lozano-Perez [29] use the same technique to find the rotation of planar features. In their method, $\mathbf{r}$ is evaluated for every pair of normals $(\hat{\mathbf{n}}_i, \hat{\mathbf{n}}_j)$ and the optimal $\mathbf{r}$ is obtained using clustering methods.

Least Square Solution using Singular Value Decomposition

Arun et. al. [3] use a representation of the measurements $\{\mathbf{v}_i'\}$ and of the model features $\{\mathbf{v}_i\}$ relative to their center of mass. In their method, the optimal estimate of rotation $\hat{R}$ under the L.S. criterion is given by the minimum over

$$E = \sum_i \|\mathbf{v}_i' - R\mathbf{v}_i\|^2$$

subject to $RR^t = I$. It can be shown [3] that this equation is equivalent to minimizing $E = \text{trace}(RH)$ where $H = \sum_i \mathbf{v}_i{}^t \mathbf{v}_i'$. Arun et. al. show that if the SVD [44] of H is $H = UVW^t$ then the minimum of E is obtained when $R = WU^t$.
Haralick et. al. [30] arrive at the same solution where they enforce the orthogonality constraints on R through Lagrange Multipliers, i.e. they minimize the following equation:

$$E = \sum_i \|\mathbf{v}_i' - R\mathbf{v}_i\|^2 + \|\Lambda(RR^t - I)\|^2$$

where Λ is the symmetric matrix of Lagrange Multipliers $\lambda_1, \ldots, \lambda_6$.

Least Square Solution using Orthonormal Matrices

Another method for finding the optimal R which minimizes $E = \text{trace}(RH)$ (described above) is presented by Horn et. al. [38] using a decomposition of the matrix H; If H is non-singular it can be decomposed into a product of an orthonormal matrix W and a symmetric matrix S:

$$H = WS$$

where $S = (H^tH)^{1/2}$ and $W = H(H^tH)^{-1/2}$. Horn et. al. show that the optimal rotation which minimizes E is $R = W$.

2.4.2 Iterative Solutions

Iterative method using Euler Angles

Lin et. al. [48] suggest an iterative solution to find the rotation where, here too, the coordinates of the points are relative to the center of mass. They show that if the points are co-planar and the rotation is normal to the plane (for example, when the points are on the x-y plane and the rotation is about the z axis) then the rotation angle θ is given by:

$$\tan(\theta) = \frac{\sum_i \mathbf{v}_i \times \mathbf{v}'_i}{\sum_i \mathbf{v}_i \cdot \mathbf{v}'_i}$$

This results is extended to solve for rotation in the general case. The general rotation is represented using Euler Angles:

$$R = R_y(\psi)R_x(\phi)R_z(\theta)$$

At each step, two of the three Euler Angles are assumed constant and the above equation is used to find the third angle. The process sequentially evaluates the three angles and the process is repeated iteratively until the solution converges.

Iterative Weighted Least Squares

In the previously described methods, the optimal solution was chosen under the L.S. criterion in which the objective function f was quadratic in the residual error, i.e. $f(\varepsilon_i) = \varepsilon_i^2$, where $\varepsilon_i = \|(Rp_i + \mathbf{t} - \mathbf{p}'_i)\|$. The problem in using this estimator is that a small number of outliers can pull the solution away from the true solution [40].
Haralik et. al. [30] suggest a robust approach based on M-estimation. In their method, the objective function is not quadratic in the residual error but represented by the Tukey function [40]. This function gives a quadratic penalty in a given close range and constant penalty elsewhere. The suggested algorithm is iterative: at each step a set of equations are solved under the W.L.S. criterion (weighted least squares), i.e. $R, \mathbf{t}$ are found that minimize:

$$E = \sum_i w_i \|R\mathbf{p}_i + \mathbf{t} - \mathbf{p}'_i\|^2$$

where the weights w_i are given by the Tukey function and dependent on the current residuals. The process continues iteratively until the solution converges.

2.5 Rigid Case: Methods for Pose Estimation from 2D Data

In contrast with the case in which 3D feature points are used as measurements, when the input measurements are 2D they also depend on the sensor model (e.g. the camera model when the measurements are taken from an image). There are two commonly used projection models for describing the transformation of a point from 3D coordinates to 2D coordinates: the *perspective projection* model and the *orthographic projection* (weak perspective) model [36].

The perspective projection model is described by a projective transformation from $\mathcal{P}^3$, which is a three dimensional projective space, onto the two dimensional projective space $\mathcal{P}^2$, where the location coordinates in $\mathcal{R}^3$ and in $\mathcal{R}^2$ are given in homogeneous coordinates. This transformation is given by:

$$M \cdot (x, y, z, 1)^t = (u, v, w)^t$$

where M is a 3x4 matrix and where $(x, y, z, 1)$, (u, v, w) are the homogeneous coordinates of the model point and the image point, respectively. The matrix M contains 12 parameters which are dependent on the positioning of the camera with respect to the object's coordinates (the extrinsic parameters) and on the camera parameters (the intrinsic parameters) such as focal length, piercing point, scaling factor etc. [83, 72]. In this work we assume the camera parameters are given and estimate only the extrinsic parameters (this problem is known also as the exterior orientation problem). Given the matrix M, the extrinsic parameters can be derived, i.e the rotation R and the translation $\mathbf{t}$ [83, 72]. In fact, when the scaling factor and focal length are both equal to one, the matrix M is given as:

$$M = \left(\begin{array}{cc} R & \mathbf{t} \end{array} \right)$$

Estimating the matrix M requires dealing with two non-linear constraints which are due to the orthogonality of the rotation R [83]. This model describes the image obtained by a pinhole camera and is a good approximation of the image obtained by most modern cameras [83].

The orthographic projection (weak perspective) is described by an affine transformation from $\mathcal{R}^3$ to $\mathcal{R}^2$ which includes a projection, as follows:

$$\Pi(s \cdot R(x, y, z)^t + \mathbf{t}) = (u, v)^t$$

where $(R, \mathbf{t})$ are the rotation and translation, s is the scaling factor and Π represents orthographic projection (i.e. the third coordinate is eliminated). The orthographic projection model gives a good approximation of measurements taken from images only when the size of the object is small relative to its distance from the camera [10]. When this condition is not satisfied, this projection model is not precise since it does not take into consideration the perspective distortions created in the image.

2.5.1 Closed Form Solutions

In contrast with the case in which 3D feature points are used, in the case where 2D data are used, the closed form solutions which assume rigidity of the transformation are applicable only to a limited number of measurements (see for example [24]). When a solution is to be obtained from a large number of measurements, no closed form solution is found so far and the methods to be found are iterative [11, 45]. When the constraints are not considered, the transformation can be found, as in the 3D case, using a direct L.S. solution for the overdetermined system of linear equations (see [11, 74, 25] for examples using orthographic projection, and [49, 83] for perspective projection). Using a direct linear L.S. solution, the 12 parameters of matrix M are estimated as independent parameters. As in the 3D methods, these methods which do not take into account the rigidity constraints, will give good solutions only in the ideal case when there is no noise in the measurements.

Several methods solve the problem analytically when few measurements are given and when the model points are in a specific configuration. Fischler and Bolles [24], for example, give a closed form solution for three or four co-planar model points. They derive the solution by first evaluating the "legs" which are the lengths of the segments connecting the camera focal point and the 3D points. From these, they derive the extrinsic parameters. Note that this analytic solution for a minimal number of points is given in the context of the Random Sample Consensus (RANSAC) algorithm which recalculates

the transformation for every three randomly chosen measurements until the calculated transformation is compatible with a large enough number of measurements. Thus, this algorithm is in essence, an iterative algorithm.

Although analytical solutions using a small number of measurements are fast and do not require a-priori guess, they do require predesign of the model points (for example that the 4 chosen model points are coplanar). Additionally, since these methods are based on a small number of points, they are very sensitive to noise in the measurements.

2.5.2 Iterative Solutions

Most of the methods for estimating the rigid transformation given 2D data, are iterative. These methods can deal with any number of measurements and most of them find the optimal solution under the L.S. criterion using non linear L.S. techniques. Haralick et.al. [30] present three iterative methods for solving the problem:

Iterative L.S. Solution: This method assigns an initial length to each "leg" l_k connecting the focal point with a point k. The 3D coordinates of point k, $\mathbf{p}'_k$, relative to the camera frame of reference, are calculated from the lengths l_k and from the corresponding 2D measurements given as coordinates in the image plane:

$$\mathbf{p}'_k = \frac{l_k}{f}(u_k, v_k, 1)$$

where (u_k, v_k) are the coordinates in the image plane and f is the focal length. The problem is now reduced to the pose estimation problem from 3D data. Using one of the techniques described in Section 2.4, an optimal estimate for the transformation (the rotation R and the translation $\mathbf{t}$) can be found. The estimated transformation is used to recalculate the legs $\{l_k\}$. The above process is repeated until the solution converges.

L.S. Adjustment by Linearization: In this method the projective matrix M described in Section 2.5 is rewritten so that the parameters describing the rotation are the three Euler angles θ, ϕ, ψ. The dependence of the projection on these parameters is non linear (see [30]). The process is initialized by assigning initial values to the three angles θ, ϕ, ψ and to the translation vector $\mathbf{t}$. These initial values converge to the final

solution using Newton iterations [71]. At each step correction values for the parameters ($\delta\theta, \delta\phi, \delta\psi$ and $\delta\mathbf{t}$) are evaluated and the process continues until the correction values converge to zero.

Robust M-Estimation: The two iterative methods mentioned above can become more robust by assigning appropriate weights to every equation obtained from a given measurement. The solution of the equations will then be performed under the weighted L.S. criterion. The weights assigned to each equation are not held constant but depend on the solution obtained at the previous iteration, i.e. they are dependent on the current residual error $\{\epsilon_i\}$:

$$\epsilon_i = M\mathbf{p}_i - \mathbf{q}_i$$

where M is the projection matrix obtained from the previous iteration. $\mathbf{p}_i$ is the 3D vector of coordinates of the model point and $\mathbf{q}_i$ is the 2D coordinates of the measurement point. The weights that Haralick et.al. suggest are derived from the Huber objective function or from the Tukey objective function [40].

Yuan [81] suggests another iterative method for solving the exterior orientation problem. As in the first method suggested by Haralick et.al. [30], Yuan evaluates both the rotation R and the lengths l_k of the legs, simultaneously. The lengths of the legs together with the measurements in the image plane define the 3D location of a point (relative to the camera frame of reference). Thus, here too, the rotation can be decoupled from the translation by considering the difference between two measurements:

$$\mathbf{p}'_k - \mathbf{p}'_j = \frac{l_k}{f}(u_k, v_k, 1) - \frac{l_j}{f}(u_j, v_j, 1) = R(\mathbf{p}_k - \mathbf{p}_j)$$

After evaluating the rotation, the translation can be calculated as explained in Section 2.4.1. Using this method, the evaluation of the rotation R, subject to the 6 rigidity constraints, requires finding the common roots of six quadratic equations. This is done iteratively using the Newton Method [71].

A method for decoupling the rotation from the translation, is also suggested by Liu et.al. [49], for the case where the measured feature are lines. Every line is represented in the object coordinate system by a unit vector $\hat{\mathbf{n}}$ defining the direction of the line in 3D. Thus the direction of the line in the camera coordinates is $R\hat{\mathbf{n}}$. Liu et.al. show that

when the line measured in the image plane is given as: $au + bv + c = 0$ where (u, v) are coordinates in the image plane, then the following equality holds:

$$R\hat{\mathbf{n}} \cdot (a, b, c)^t = 0$$

Such an equation, which is linear in the 9 parameters of R, is created for every line measurement. In order to eliminate the rotational constraints, the rotational parameters are represented using the three Euler angles. The above equation then becomes non linear in the three rotational parameters. The Euler angles are evaluated under the L.S. criterion using Newton iterations [71]. This method is also applicable when the measure features are points rather than lines since every two points define a line. Using this method at least six points or at least eight lines must be measured in order to obtain a unique solution [49].

2.6 Constrained Case: Methods for Pose Estimation

Extensive studies can be found in the literature dealing with pose estimation of rigid objects from measurements, however, little attention has been given to pose estimation of articulated or constrained objects. Several studies can be found that deal with special cases of constrained objects, namely, articulated objects having prismatic or revolute joints, most of them in recognition context ([12, 27, 26, 52, 51, 57, 66]). In general, the existing methods dealing with this problem can be divided into two main paradigms:

1. **Divide and conquer methods**
   The basic and naive method is to decompose the object into parts and to estimate the pose of each part separately. Grimson [27, 26] follows this paradigm in order to identify a family of objects which differ in scale-factor, stretch factor or the angles between parts. In his study, estimating the pose of each part separately was followed by an assessment of the current interpretation of the parts by testing whether the parts satisfy the constraints defined between them (up to a predefined threshold). Grimson uses the pose estimate of a part to limit the possible matchings of the neighboring part, however this estimate is not used in the pose estimation

of the neighboring part. Shakunaga [66] follows a similar method for estimating the orientation of a flexible body composed of parts joined together by revolute joints. Although the simplicity of this method is attractive, it is obvious that it is unsatisfying from several aspects: Evaluating the pose of each part separately may result in constraints not being satisfied between the parts. Furthermore, no mutual information passes between parts and each object component is located using only its measurements. Additional information which can be obtained from measurements of neighboring components is not considered, thus, not all available information is exploited.

2. **Parametric methods**

   It is possible to eliminate the defined constraints by decreasing the number of parameters that describe the pose of the object (so that the number of free parameters equals the degrees of freedom of the object). The remaining parameters are estimated during the estimation process. For example, the pose of an articulated object in 2D having two components connected by a revolute joint, can be described by the translation and the orientation of each component (6 parameters) with an additional constraint due to the joint between the components (Figure 2.1 left). Alternatively, the pose of the object can be described by the translation and orientation of one of the components and the relative angle to the second component (4 parameters) (Figure 2.1 right). The latter description eliminates the need to consider a constraint during the estimation process.

   Lowe [51, 52] follows this method and estimates the free parameters of the viewpoint and of the model using Newton iterations into which additional techniques are incorporated in order to ensure convergence. A similar method was used by Brooks [17] in the well known system *ACRONYM*. Mulligam et. al. [57] use the same approach for estimating the positions of an excavator's arms, however, in their work, the flow of information about pose from one arm to the next is in one direction, thus the pose of the boom influences the pose of the bucket but not vice versa.

   The main problem in the method of parameter reduction is the need for defining the dependence of each measurement on all the free parameters during the estimation process. The definition of the dependence is problematic for two reasons:

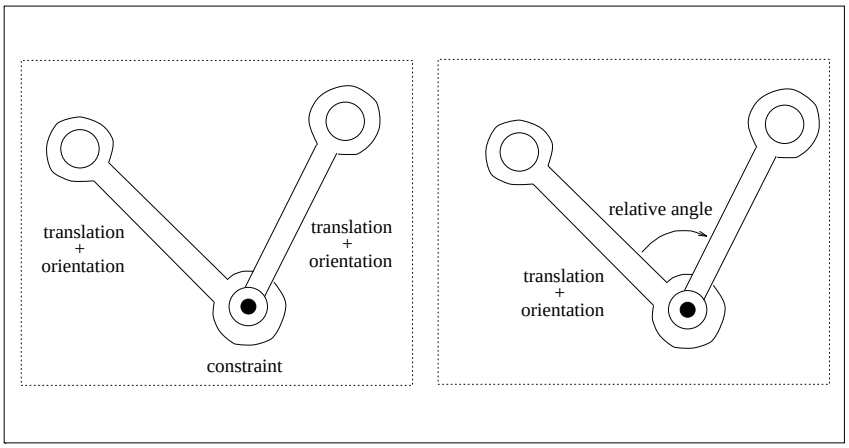

Figure 2.1: Two possibilities of parameter sets for the pose definition of an articulated object having a revolute joint.

First, the complexity of this definition increases with the number of free parameters that each point is dependent on. Second, in most cases, as the number of components of the object is greater, the order of the nonlinearity of the dependence equations is higher. This results in a more complex and less stable solution especially when using iterative methods based on linear approximation of the nonlinear equations. An additional drawback of this method is the difficulty in finding the correct free parameters that ensure that the model constraints are satisfied for any given values of these parameters. The difficulty of selecting the parameters increases with the number of constraints and with the number of parts participating in each constraint. Therefore the existing methods deal with ad-hoc techniques which handle common constraints such as revolute and/or prismatic joints where the parameters reduction is simple and immediate.

Chapter 3

Parameter Estimation

Most of the problems discussed in this dissertation can be formalized as parameter estimation problems. Parameter estimation deals with the estimation of particular unknowns from measurements where the measurements and the unknowns are functionally related. Generally, these measurements are contaminated by noise. Suppose we are interested in evaluating the parameter vector $\mathbf{s}$ from noisy measurements represented by the vector $\hat{\mathbf{z}}$ and we are given the posterior distribution $P(\mathbf{s}|\hat{\mathbf{z}})$ which represents the probability distributed function (p.d.f.) of $\mathbf{s}$ given $\hat{\mathbf{z}}$. The estimate $\hat{\mathbf{s}}$ of $\mathbf{s}$ is calculated such that some criterion will be fulfilled, i.e. $\hat{\mathbf{s}}$ is chosen to be optimal in a certain sense. For example, consider the estimate $\hat{\mathbf{s}}$ which minimizes the following expression:

$$E\{\|\mathbf{s}-\hat{\mathbf{s}}\|^2|\hat{\mathbf{z}}\} = \int_{-\infty}^{+\infty} \|\mathbf{s}-\hat{\mathbf{s}}\|^2 P(\mathbf{s}|\hat{\mathbf{z}})d\mathbf{s} \tag{3.1}$$

The estimate $\hat{\mathbf{s}}$ minimizing Eq. 3.1 is a *Bayesian Estimate* which satisfies the minimum-variance criterion and can be shown to satisfy $\hat{\mathbf{s}} = E\{\mathbf{s}|\hat{\mathbf{z}}\}$ for any distribution of $\mathbf{s}$ [19]. If the estimate $\hat{\mathbf{s}}$ is a linear function of the measurements $\hat{\mathbf{z}}$ and satisfies the minimum in Eq. 3.1, then $\hat{\mathbf{s}}$ is called the *Linear Minimum Variance Estimator* of $\mathbf{s}$. Other criteria can also be used (see [58] for further reference).

3.1 The Kalman Filter for Linear System Models

Estimation problems in stochastic systems is widely studied in the literature [43, 55, 58]. In stochastic systems a parameter vector $\mathbf{s}(t)$, called a *state vector*, describes the behavior

of the system at time t. In a general stochastic system the behavior of the system changes with time, thus the state vector is a dynamic parameter vector. However, for our purposes we deal with a degenerated case of stochastic systems where the state vector is static and does not change with time. Therefore, we denote the state vector by the static parameter vector $\mathbf{s}$.

The general *measurement model* of a static stochastic system is:

$$\hat{\mathbf{z}} = f(\mathbf{s}) + \varepsilon \quad ,$$

where $f(\mathbf{s})$ is a mathematical function of the state vector which can be measured, $\hat{\mathbf{z}}$ is the vector of actual measurements, and ε is the measurement noise, whose covariance is assumed known. By convention, actual measurements are denoted with hats while 'true' measurements (without the noise vector ε) are written without the hat. The central problem in the theory of such systems is to *estimate* the state vector $\mathbf{s}$ from the measurements. The estimate is denoted by the vector $\hat{\mathbf{s}}$.

The main tool we use to estimate a state vector is the *Kalman filter* (K.F.). The Kalman filter is a tool for estimating the state vector of a stochastic **linear** system from a sequence of measurements. Here we briefly describe the Kalman filter and state some of its properties. For a complete discussion see e.g. [43, 55, 2].

The static Kalman filter is based on the sequential measurement model:

$$\hat{\mathbf{z}}_k = H_k \mathbf{s} + \varepsilon_k$$

where this measurement is taken at time step k, $k = 1, 2, \cdots$. The matrix H_k is a linear operator relating the state vector $\mathbf{s}$ to the true measurement at time k. ε_k is the noise vector associated with the actual measurement $\hat{\mathbf{z}}_k$. It is assumed that the measurement noise is an uncorrelated zero mean vector and that its covariance matrix Λ_k is given, i.e,

$$E\{\varepsilon_k\} = 0 \quad ; \quad \mathrm{var}\{\varepsilon_k\} = \Lambda_k \quad ; \quad \mathrm{cov}\{\varepsilon_k, \varepsilon_j\} = 0 \ \text{ for } \ k \neq j.$$

Assuming that an *a priori* estimate of the state vector and its associated covariance matrix, are known from the previous time step $k-1$:

$$E\{\mathbf{s}\} = \hat{\mathbf{s}}_{k-1} \quad ; \quad \mathrm{var}\{\mathbf{s}\} = E\{(\mathbf{s} - \hat{\mathbf{s}}_{k-1})(\mathbf{s} - \hat{\mathbf{s}}_{k-1})^t\} = \Sigma_{k-1}$$

the Kalman filter updating equations for estimating $\mathbf{s}$ and Σ are:

$$\begin{aligned} &\textit{state estimate update}: && \hat{\mathbf{s}}_k = \hat{\mathbf{s}}_{k-1} + K_k(\hat{\mathbf{z}}_k - H_k\hat{\mathbf{s}}_{k-1}) \\ &\textit{state covariance update}: && \Sigma_k = \Sigma_{k-1} - K_k H_k \Sigma_{k-1} \\ &\textit{Kalman gain matrix}: && K_k = \Sigma_{k-1}H_k^t(H_k\Sigma_{k-1}H_k^t + \Lambda_k)^{-1} \quad . \end{aligned} \tag{3.2}$$

A variant formulation of the K.F. equations is [2]:

$$\hat{\mathbf{s}}_k = \Sigma_k(H_k^t\Lambda_k^{-1}\hat{\mathbf{z}}_k + \Sigma_{k-1}^{-1}\hat{\mathbf{s}}_{k-1}) \tag{3.3}$$

$$\Sigma_k = (H_k^t\Lambda_k^{-1}H_k + \Sigma_{k-1}^{-1})^{-1} \quad . \tag{3.4}$$

The Kalman filter can be abstractly described by a black box which, at each time step k, receives three inputs and gives one output (see Fig. 3.1). The inputs are:

1. **The a priori estimate input** - the *a priori* estimate of the state vector and its associated covariance matrix: $(\hat{\mathbf{s}}_{k-1}, \Sigma_{k-1})$.

2. **The measurement input** - a new measurement and an associated covariance matrix representing the noise in the measurement: $(\hat{\mathbf{z}}_k, \Lambda_k)$.

3. **The measurement model input** - a linear relationship between the current measurement and the state vector:

   $$\mathbf{z}_k = H_k\mathbf{s}$$

   where $\mathbf{z}_k$ stands for the real measurement s.t. $\hat{\mathbf{z}}_k = \mathbf{z}_k + \varepsilon_k$.

The output of the Kalman filter box is a posterior estimate of the state vector and its associated covariance matrix: $(\hat{\mathbf{s}}_k, \Sigma_k)$. The posterior estimation is "better" than the *a priori* estimate, i.e. $\Sigma_k \leq \Sigma_{k-1}$ (if $\Sigma_k < \Sigma_j$ then $\Sigma_j - \Sigma_k$ is positive definite).

The K.F. equations yield an unbiased estimate of $\mathbf{s}$ which is optimal under the linear minimal variance criterion [2]. In the case where the measurement noise ε_k is a Gaussian process (which is a reasonable assumption, considering the numerous sources of noise), the K.F. gives an estimate which is also optimal in the sense of the minimum variance and

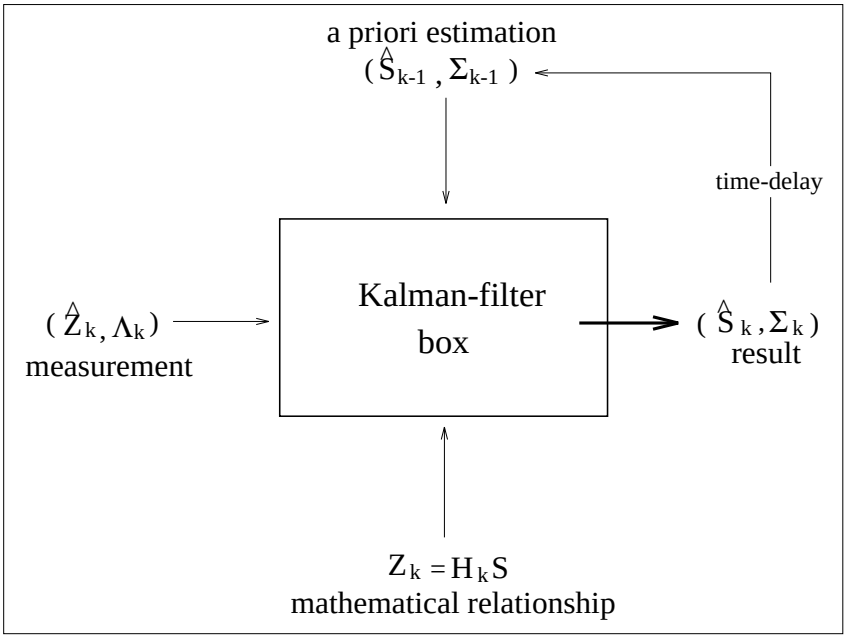

Figure 3.1: The Kalman filter for static-parameter estimation; The three inputs and the estimation output.

the maximum-likelihood criterion [2]. The estimate also coincides with the conditional expectation:

$$\hat{\mathbf{s}}_k = E\{\mathbf{s}|\hat{\mathbf{z}}_1, \hat{\mathbf{z}}_2 \cdots \hat{\mathbf{z}}_k\} \quad .$$

As stated before, the general K.F. deals with a parameter vector that changes with time, whereas in our case the state vector $\mathbf{s}$ is static and does not change between measurements. Therefore, our case is a degenerate K.F. and we call its fusion process a *K.F. fuser* since it fuses the current and the *a priori* information in order to obtain a better estimate of the state vector.

Note that the estimation obtained using the K.F. equations is equivalent to that obtained by weighted least-squares methods [71] using the optimal weight Λ_k^{-1} (i.e. the expression $\sum_{i=1}^{k}(\mathbf{z}_i - H_i\mathbf{s})\Lambda_i^{-1}(\mathbf{z}_i - H_i\mathbf{s})^t$ is minimal for $\hat{\mathbf{s}}_k$). However in this thesis we prefer to deal with the estimation problems in terms of the K.F. The advantages in using the K.F. method will be described later.

3.2 The Extended Kalman Filter

The standard K.F. equations are applicable to linear systems, i.e. where the measurements are linearly dependent on the state-vector: $\mathbf{z}_k = H_k\mathbf{s}$. In the general case, the measurements are functionally related to the state vector through a non-linear implicit vectorial function:

$$h_k(\mathbf{s}, \mathbf{z}_k) = \mathbf{0}$$

In this case we use the *Extended Kalman filter* (E.K.F.) process which is a generalization of the *Kalman Filter* (K.F.) to non-linear systems [43, 55]. The transition from step $k-1$ to step k is performed using a linear approximation of h_k by taking the first order Taylor expansion around $(\hat{\mathbf{s}}_{k-1}, \hat{\mathbf{z}}_k)$:

$$h_k(\mathbf{s}, \mathbf{z}_k) = 0 \approx h_k(\hat{\mathbf{s}}_{k-1}, \hat{\mathbf{z}}_k) + \frac{\partial h_k}{\partial \mathbf{s}}(\mathbf{s} - \hat{\mathbf{s}}_{k-1}) + \frac{\partial h_k}{\partial \mathbf{z}}(\mathbf{z}_k - \hat{\mathbf{z}}_k) \tag{3.5}$$

Equation 3.5 can be rewritten as a linear equation:

$$\tilde{\mathbf{z}}_k = \tilde{H}_k\mathbf{s} + \tilde{\varepsilon}_k \quad , \tag{3.6}$$

where

$$\begin{aligned} \tilde{\mathbf{z}}_k &= -h_k(\hat{\mathbf{s}}_{k-1}, \hat{\mathbf{z}}_k) + \frac{\partial h_k}{\partial \mathbf{s}}\hat{\mathbf{s}}_{k-1} \\ \tilde{H}_k &= \frac{\partial h_k}{\partial \mathbf{s}} \\ \tilde{\varepsilon}_k &= \frac{\partial h_k}{\partial \mathbf{z}}(\mathbf{z}_k - \hat{\mathbf{z}}_k) \quad . \end{aligned}$$

and where the derivetions are taken at the point $(\hat{\mathbf{s}}_{k-1}, \hat{\mathbf{z}}_k)$. $\tilde{\mathbf{z}}_k$ represents the current "measurement"and $\tilde{H}_k$ is the matrix denoting a linear connection between the "measurement" and the state vector $\mathbf{s}$. The term $\tilde{\varepsilon}_k$ depicts the noise in the "measurement" $\tilde{\mathbf{z}}_k$ and satisfies:

$$E\{\tilde{\varepsilon}_k\} = 0 \quad ; \quad \mathrm{var}\{\tilde{\varepsilon}_k\} = (\frac{\partial h_k}{\partial \mathbf{z}})\Lambda_k(\frac{\partial h_k}{\partial \mathbf{z}})^t \quad ; \quad \mathrm{cov}\{\tilde{\varepsilon}_k, \tilde{\varepsilon}_j\} = 0 \quad \forall\ k \neq j \quad .$$

The standard K.F. updating equations can be applied to the linearized system, as explained above.

The linear approximation of h_k produces imprecisions when linearization is performed around a point that is not close enough to the true state vector. In order to reduce the influence of the linearization *local iterative K.F.* [55, 43] is applied. During the iterations the non-linear measurement equation is relinearized around the updated state estimate obtained from the K.F. updating equations, and another cycle of K.F. is performed using a new version of the linearized equation. These iterations increase the time complexity of the solution.

3.3 Why Do We Use the Kalman Filter?

Although there is no computational advantage in using the K.F. over other methods such as weighted least squares error methods, we find it advantageous to use the K.F. in this thesis because of the following reasons:

- The K.F. takes into account, explicitly, the uncertainties of the measurements.
- The K.F. supplies an uncertainty associated with the resulting estimate and thus provides an evaluation of the quality of the estimate.
- Given a new measurement, the K.F. enables recursive updating of the estimate. There is no need of repeating the full estimation process and there is no need of replicating nor retaining previous measurements. Only the last estimate and its uncertainty are required for updating.
- Methods and techniques developed for the K.F. can be used, such as methods dealing with non-linearities (E.K.F.) and methods for increasing numeric stability (these methods will be described later).
- The recursive feature of the K.F. is in accord with the need of a sequential pose estimation process when the correspondence between the model points and the measured points is not given (see Section 1.3).

3.4 Physical Interpretation of the Kalman Filter

An additional advantage in using the K.F. is the ability to give a physical interpretation to the solution. The physical interpretation consists of a mechanical system whose physical solution is the solution obtained by the K.F. equations. The physical motivation stems from two main reasons:

- The solution of the mechanical system is derived from existing physical laws. The equality between this solution and the solution obtained from the K.F. equations, suggests that there is a basis for a natural optimality criterion in the solution of the K.F. and that this criterion is not arbitrary.

- The mechanical system supplies us with intuition and insight on the behavior of the system and its solution.

For example, assume the state vector we estimate represents the location of a point P in a two dimensional plane, i.e. $\mathbf{s} = (x, y)$. Further assume that n measurements of the point location were obtained: $\{(\hat{\mathbf{z}}_i, \Lambda_i)\}_{i=1...n}$ where $\hat{\mathbf{z}}_i$ is a 2D vector representing the measurement value in the x-y plane and Λ_i represents its uncertainty. We can represent this estimation problem by an equivalent physical system where each measurement $(\hat{\mathbf{z}}_i, \Lambda_i)$ is represented by a spring (having length equal to zero) which pulls the point P towards the location $\hat{\mathbf{z}}_i$ and which has a spring constant K_i inversely proportional to Λ_i, i.e. $K_i = \Lambda_i^{-1}$ (see Fig. 3.2).

The solution for the physical system is that which brings the energy to a minimum. We show that the solution resulting from the K.F. equations is equivalent to the physical solution. The K.F. solution is the output obtained given the following three inputs:

1. the *a priori* estimate input - since no *a priori* knowledge can be assumed about the location of point P, we take $(\hat{\mathbf{s}}_0, \Sigma_0)$ as any initial estimate value $\hat{\mathbf{s}}_0 = (\hat{x}_0, \hat{y}_0)^t$ and associate with it an infinite uncertainty Σ_0.

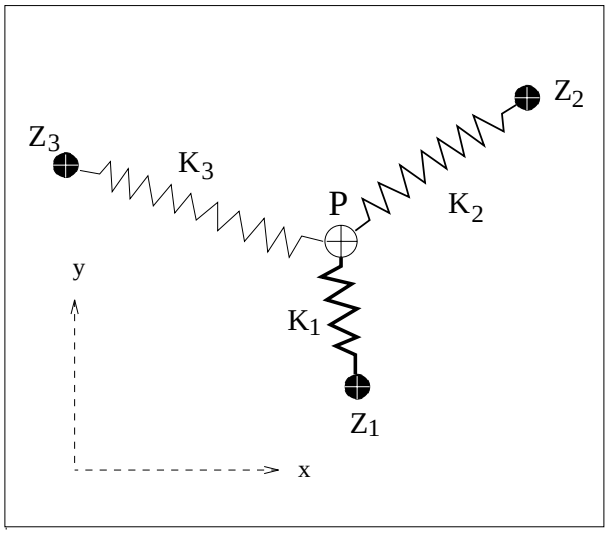

Figure 3.2: A physical system equivalent to an estimation problem: the 2D pose of point P is determined by 3 measurements. Each measurement i is represented by a zero-lengthed spring which pulls the point P towards $\hat{\mathbf{z}}_i$ and which has a spring constant K_i inversely proportional to the uncertainty of the measure.

2. the measurement input - is taken as:

$$\left[\begin{pmatrix} \hat{\mathbf{z}}_1 \\ \hat{\mathbf{z}}_2 \\ \vdots \\ \hat{\mathbf{z}}_n \end{pmatrix}, \begin{pmatrix} \Lambda_1 & & 0 \\ & \ddots & \\ 0 & & \Lambda_n \end{pmatrix}\right] = [\hat{\mathbf{z}}, \Lambda]$$

3. the model measurement - is taken as $\mathbf{z} = H\mathbf{s}$ where $H = (\overbrace{I, I, \cdots I}^{n \text{ times}})$.

The output obtained from the K.F. updating equations given these inputs is (see Section 3.1):

$$\hat{\mathbf{s}}^{\text{k.f.}} = (\sum_i^n \Lambda_i^{-1})^{-1} \sum_i^n \Lambda_i^{-1} \hat{\mathbf{z}}_i$$

$$\Sigma^{\text{k.f.}} = (\sum_i^n \Lambda_i^{-1})^{-1}$$

The solution to the physical system will give a location $\hat{\mathbf{s}}^{\text{phy}}$ such that the sum of forces acting on P is zero, i.e.:

$$F = \sum_i^n F_i = \sum_i^n K_i(\hat{\mathbf{s}}^{\text{phy}} - \hat{\mathbf{z}}_i) = 0$$

or

$$\hat{\mathbf{s}}^{\text{phy}} = (\sum_i^n K_i)^{-1} \sum_i^n K_i \hat{\mathbf{z}}_i = (\sum_i^n \Lambda_i^{-1})^{-1} \sum_i^n \Lambda_i^{-1} \hat{\mathbf{z}}_i = \hat{\mathbf{s}}^{\text{k.f.}}$$

Thus, the solution to the physical system is equivalent to the solution obtained from the K.F. equations. If we replace the set of springs (measurements) with a single equivalent spring having a spring constant K^{tot} and anchored at $\hat{\mathbf{s}}^{\text{phy}}$ we obtain:

$$F = K^{\text{tot}} \Delta\mathbf{s} = \sum_i^n K_i(\hat{\mathbf{s}}^{\text{phy}} - \hat{\mathbf{z}}_i + \Delta\mathbf{s})$$

but since $\sum_i^n K_i(\hat{\mathbf{s}}^{\text{phy}} - \hat{\mathbf{z}}_i) = 0$ we have $K^{\text{tot}}\Delta\mathbf{s} = \sum_i^n K_i \Delta\mathbf{s}$ and then

$$K^{\text{tot}} = \sum_i^n K_i = \sum_i^n \Lambda_i^{-1} = (\Sigma^{\text{k.f.}})^{-1}$$

That is, the equivalent spring constant K^{tot} is inversely proportional to the uncertainty matrix of $\hat{\mathbf{s}}^{\text{k.f.}}$. Thus the quality of the estimation obtained from the K.F. equations is equivalent to the "strength"of the equivalent spring in the physical system. It is now obvious that the effect of adding an additional measurement to the system is equivalent to adding an additional spring to the physical system. In such cases, the spring constant of the equivalent spring always increases, thus in the K.F. system $\Sigma_{t+1} \leq \Sigma_t$. Additionally, as the number of springs in the system increases, the effect of adding another measurement, decreases. The corresponding analogy in the K.F. system is that as more measurements are added, they have a smaller effect on the solution.

3.5 The Complexity of the Kalman Filter

Every application of the K.F. fuser requires an inversion of a d-dimensional matrix where d is the dimension of the measurement vector. Practical algorithms for matrix inversion take $O(d^3)$. The remaining matrix multiplication operations in the K.F. equations take

$O(nd^2)$ or $O(n^2d)$ where n is the dimension of the state vector. Thus, if d is relatively larger than n, the overall complexity is $O(d^3)$ and when d is small compared to n we have an overall complexity of $O(n^2d)$. In the case where the measurement model input is non-linear, local iterations are used where each iteration takes max[$O(d^3)$,$O(n^2d)$]. It can be shown that the iterative process is equivalent to solving a set of non-linear equations using Newton iterations where the solution at each step is evaluated using the weighted least squares criterion. Thus the convergence rate of the iterations in the K.F. process is equivalent to that of the Newton method which is quadratic near the solution [71].

3.6 The Estimation Criterion

As stated in Section 1.2, the measurement noise is modeled, in this thesis, by a Gaussian process with known parameters. This is reasonable due to the large number of noise sources. However, the Gaussian model does not always give a good representation of the measurements; in many cases the distribution is better modeled by a Gaussian-like distribution with an additional tail-distribution (outliers) because of gross-error measurements (which have large deviations from the average). These gross-error measurements have a large influence on the estimate solution especially when the solution is based on the minimum variance criterion (which gives an estimate that has a breakdown point $\varepsilon^* = 0\%$ [40]). Several methods have been developed to give robust solutions to this problem [30, 24, 45]. These methods use estimators based on criteria other than the minimum variance (such as minimum absolute value) and have been applied to pose estimation problems. Despite the above described drawbacks, we chose to work with a Gaussian noise model and with the minimum variance criterion (Kalman filter) due to the following reasons:

- The robust methods do not deal with the uncertainty of the measurements in an explicit manner (using the covariance matrix), but combine them into a single value (the trace of the covariance matrix) representing the quality of the contribution of each measurement to the final solution. This contraction into a single value causes loss of information, which in our work may have severe significance in the cases where measurements have infinite uncertainty in one direction and a small

uncertainty in another (this happens, for example, when the measurements are taken from a projection).

- The robust methods are not easy to analyze analytically and so do not have closed-form estimators.

- Addition of information or inclusion of *a priori* information to improve the solution, is easily performed using minimum variance based methods (see K.F equations). It is not easily done using the robust methods, since these require re-evaluation of all the data. Thus robust methods also require larger memory space.

- One can improve minimum variance based methods to be more robust by eliminating outlier measurements by applying a goodness of fit test. These tests will extract those measurements whose Mahalanobis distance between prediction and measurement value is greater than a given value (see Appendix A and Zhang [82]).

Chapter 4

Pose Estimation of Rigid Objects

4.1 Introduction

In model-based pose determination the position of a known object is determined from different types of surface measurements (for example: [30, 49, 67, 3]). Usually feature points such as maximum curvature, segment endpoints and corners are measured. The aim of this chapter is to determine the correct pose of the object (location and orientation) relative to the sensor frame of reference where the measurements are noisy. This problem is known as *absolute orientation* in photogrametry and solution methods are classified into two major categories according to the type of measurements [30]:

1. $3D$ to $3D$ correspondence: both model and measurements give the $3D$ location of features (measurements from range data, stereo, etc.).

2. $3D$ to $2D$ correspondence: the model is $3D$ while the available measurements supply projected $2D$ information, like locations in an image plane. The projection can be either perspective or orthographic.

In this chapter we suggest a uniform framework to compute the absolute orientation, where the measured data can be a mixture of $1D$, $2D$ and $3D$ information. Unifying the different types of measurements is done by associating an uncertainty matrix with each measured feature. Uncertainty depends both on the measurement noise and on the type of measurement. This representation unifies the different categories of the

absolute orientation problem into a single problem that varies only in the uncertainty values associated with the measurements. With this paradigm we obtain a uniform mathematical formulation of the problem and can fuse different kinds of measurements to obtain a better solution. The measurements fusion is done using K.F. tools. The algorithm we describe has the additional advantages of supplying a certainty measure of the estimate, enabling an efficient matching strategy and allows simple parallelization.

4.2 Definition of the Problem

A *model* M of a $3D$ object consists of n model points whose locations relative to an object-centered frame of reference is known:

$$\mathbf{u}_i = (x_i, y_i, z_i)^t \quad \text{for each } i = 1 \cdots n$$

A *measurement* M' of a $3D$ object consists of m measurements where the location coordinate and the associated covariance matrix of each measurement are given in a viewer-centered frame of reference:

$$(\hat{\mathbf{u}}'_j, \Lambda_j) \quad \text{for each } j = 1 \cdots m$$

$\hat{\mathbf{u}}'_j$ - is a noise-contaminated measure of the real location-vector $\mathbf{u}'_j$ associated with the j^{th} measured point. It is possible to have more than one measurement for a single model point.
Λ_j - is the covariance matrix depicting the uncertainty in the sensed vector $\hat{\mathbf{u}}'_j$. We do not constrain the dimensionality of the measured data but allow it to be $3D$ (stereo, range finder etc.), $2D$ (orthographic or perspective projection) or $1D$ (uncalibrated range finder).

A *matching* between the model M and the measurement M' is a collection of pairs of the form $\{(i, j)\}$, which represents a correspondence between the i^{th} model points and the j^{th} measurement. For simplicity we further denote with the same index a model point and its matched measurement.

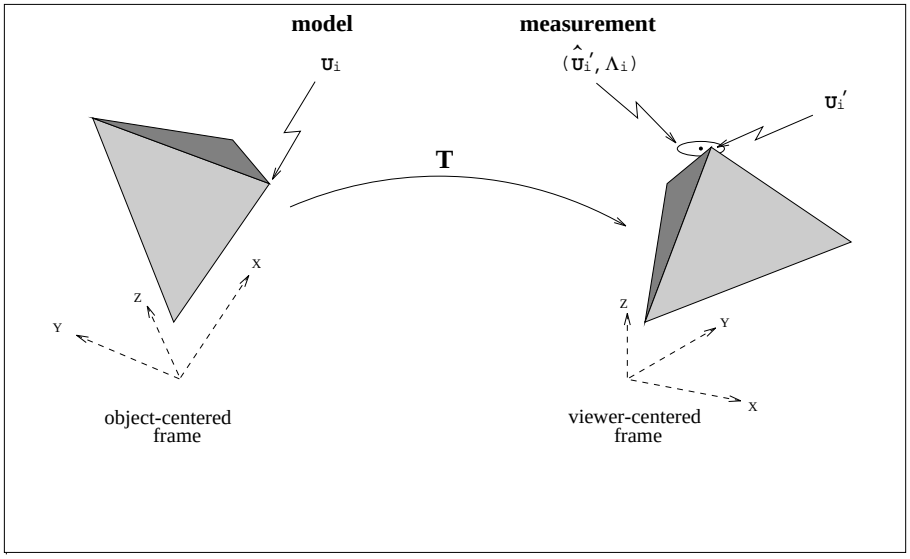

Figure 4.1: A $3D$ object and its transformation. A model point location $\mathbf{u}_i$ is represented in an object-centered frame of reference. $\mathbf{u}_i'$ is its location in a viewer-centered frame of reference. The pair $(\hat{\mathbf{u}}_i', \Lambda_j)$ is the actual measurement of $\mathbf{u}_i'$ with its associated uncertainty. $\mathbf{T}$ is the transformation from the object-centered coordinates $\mathbf{u}_i$ to the viewer-centered coordinates $\mathbf{u}_i'$.

The problem:
Given a model M, a measurement M' and a matching as above, estimate a transformation T which optimally maps the coordinates $\mathbf{u}_i$ of the model points onto the corresponding measured locations $(\hat{\mathbf{u}}_i', \Lambda_i)$. The estimated transformation T describes the position of the measured object M' in the $3D$ scene. The meaning of the optimality of the estimated transformation will be given later on.

4.3 The Measurement Unification and Its Uncertainty

As previously noted, we are given the uncertainty matrix of each and every synthesized point feature. That is, each extracted feature is associated with both estimated parameter

values given in a viewer-centered frame of reference, and an uncertainty of these values. The uncertainty is derived from several factors:

- Uncertainty due to measurement noise (e.g. digitization, blurring and chromatic aberrations).
- Uncertainty dependent upon the feature detection process. For example, a detected end-point of a line segment will have high uncertainty in the direction of the line and low uncertainty in the perpendicular direction.
- Uncertainty due to the lack of information caused by projections.

Modeling of the measurement noise is outside the scope of this thesis and we assume the measurement uncertainty is given. Using the given uncertainty, a unified representation of the measured data is constructed. In the following, the measure unification is presented. The possible types of measurements are divided into three categories:

$3D$ measured data:

The simplest case is that of a point $q \in M'$ represented by the pair:

$$q = [\hat{\mathbf{u}}', \Lambda] \quad ,$$

where $\hat{\mathbf{u}}' = (\hat{x}', \hat{y}', \hat{z}')$ is the measured location vector and Λ is its uncertainty.

$2D$ projected data:

When the measurements are obtained using a projection, the measured data is described as a measurement in $3D$ where the uncertainty in the direction of projection is infinite. Assume that the measurements are performed on the image plane using the coordinate system (v, w):

$$proj(q) = [(\hat{v}, \hat{w}), \Sigma_{vw}] \quad ,$$

where Σ_{vw} is a 2×2 covariance matrix describing the uncertainty of the measurement $(\hat{v}, \hat{w})$.

In the case where the projection is along the z-axis (orthographic), this data is represented as:

$$q = [(\hat{v}, \hat{w}, \hat{z}'), \begin{pmatrix} \Sigma_{vw} & 0 \\ 0 & \infty \end{pmatrix}] \quad ,$$

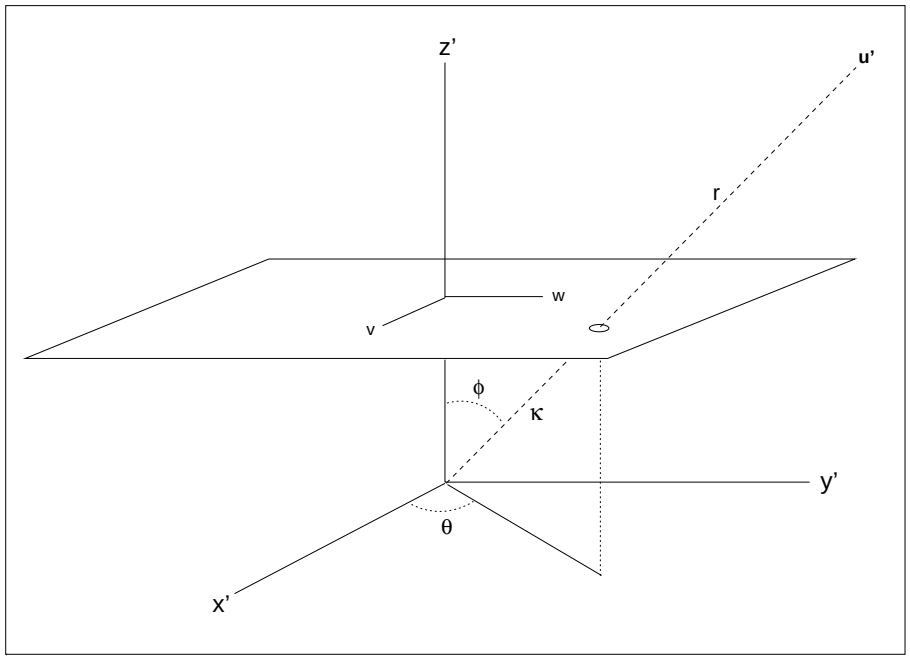

Figure 4.2: A perspective projection of the point $\mathbf{u}'$ onto an image plane (v, w). The point can be represented in either, a Cartesian system (x', y', z') or a spherical system (r, ϕ, θ).

where $\hat{z}'$ is any estimate of the z' coordinate.

In the case of a perspective projection, the modeling of the uncertainty is a little more complex. Assume that the origin of the viewer-centered frame of reference is at the focal point as shown in Figure 4.2 and the focal length is equal to one. We aim to transform the measurement given in the image-plane coordinate system into a representation in the Cartesian system (x', y', z').

Considering the spherical coordinate system (r, ϕ, θ) (Figure 4.2). The vector $(\hat{v}, \hat{w})$ determines the angular coordinates (ϕ, θ) but leaves the value of r undetermined:

$$\begin{aligned} \hat{\phi} &= \arctan(\sqrt{\hat{v}^2 + \hat{w}^2}) \\ \hat{\theta} &= \arccos(\frac{\hat{v}}{\sqrt{\hat{v}^2 + \hat{w}^2}}) \quad . \end{aligned}$$

Additionally, the uncertainty of $(\hat{v}, \hat{w})$ is translated into a covariance matrix in the (ϕ, θ) system as follows:

$$\Lambda_{\phi\theta} = \left(\frac{\partial(\phi,\theta)}{\partial(v,w)}\right) \Sigma_{vw} \left(\frac{\partial(\phi,\theta)}{\partial(v,w)}\right)^t \quad ,$$

where $\frac{\partial(\phi,\theta)}{\partial(v,w)}$ is the Jacobian of the transform from (v, w) to (ϕ, θ), and the derivative is taken at point $(\hat{v}, \hat{w})$. The Jacobian matrix is:

$$\frac{\partial(\phi,\theta)}{\partial(v,w)} = \begin{pmatrix} \hat{v}\kappa^2\psi & \hat{w}\kappa^2\psi \\ -\hat{w}\psi^2 & \hat{v}\psi^2 \end{pmatrix} \quad ,$$

where

$$\begin{aligned} \kappa &\equiv \frac{1}{\sqrt{\hat{v}^2 + \hat{w}^2 + 1}} \\ \psi &\equiv \frac{1}{\sqrt{\hat{v}^2 + \hat{w}^2}} \ . \end{aligned}$$

The transformation into spherical coordinates, as an intermediary stage, allows a simple representation of the measurement in $3D$:

$$q = [(\hat{r}, \hat{\phi}, \hat{\theta}), \Lambda_{r\phi\theta}] \ ,$$

where

$$\Lambda_{r\phi\theta} = \begin{pmatrix} \infty & 0 & 0 \\ 0 & & \\ 0 & & \Lambda_{\phi\theta} \end{pmatrix}$$

and $\hat{\phi}$, $\hat{\theta}$, $\Lambda_{\phi\theta}$ are the expressions described above. $\hat{r}$ is unknown but an estimation of $\hat{r}$ will be chosen as is explained later in this section.

In practice we are interested in representing the measurement in Cartesian coordinates, thus, the measurement is transformed again from the spherical coordinates to Cartesian coordinates (x', y', z') as follows:

$$q = [(\hat{x}', \hat{y}', \hat{z}'), \Lambda_{xyz}] \quad ,$$

where

$$\begin{aligned} \hat{x}' &= \hat{r} \sin\hat{\phi} \cos\hat{\theta} = \hat{z}'\hat{v} \\ \hat{y}' &= \hat{r} \sin\hat{\phi} \sin\hat{\theta} = \hat{z}'\hat{w} \\ \hat{z}' &= \hat{r} \cos\hat{\phi} = \kappa\hat{r} \end{aligned}$$

and the covariance matrix is:

$$\Lambda_{xyz} = \left(\frac{\partial(x,y,z)}{\partial(r,\phi,\theta)}\right) \Lambda_{r\phi\theta} \left(\frac{\partial(x,y,z)}{\partial(r,\phi,\theta)}\right)^t \quad .$$

The Jacobian is

$$\frac{\partial(x,y,z)}{\partial(r,\phi,\theta)} = \begin{pmatrix} \hat{v}\kappa & \hat{r}\hat{v}\kappa\psi & -\hat{r}\hat{w}\kappa \\ \hat{w}\kappa & \hat{r}\hat{w}\kappa\psi & \hat{r}\hat{v}\kappa \\ \kappa & -\hat{r}\kappa/\psi & 0 \end{pmatrix} \quad ,$$

where the derivative is taken at the point $(\hat{r}, \hat{\phi}, \hat{\theta})$. Here too, all values are known except for $\hat{r}$. Since the solution to the location problem incrementally improves the estimation of T, i.e. at step k there exists an estimate $\hat{T}^{k-1}$ from the previous step. We use this estimate to calculate an estimate of $\hat{r}$ at step k as follows:

$$\hat{r}^k = \|\hat{T}^{k-1}(\mathbf{u})\| \quad ,$$

where $\mathbf{u} \in M$ is the location of the corresponding point in the model. We emphasize that the uncertainty of this estimate, as expressed in the covariance matrix, is infinite.

<u>1*D* measured data:</u>

In this case only distances from the observer to the object features are given while directions are unknown. An example for such a case is a hand held range finder. Considering the same spherical coordinates (r, ϕ, θ) as in Figure 4.2, a $1D$ measurement determines the value of r but leaves the angular coordinates (ϕ, θ) undetermined. Integration of a such measurement is done in a similar fashion as in the perspective projection where in this case the spherical covariance matrix $\Lambda_{r\phi\theta}$ is:

$$\Lambda_{r\phi\theta} = \begin{pmatrix} \mathrm{var}(r) & 0 & 0 \\ 0 & \pi^2/3 & 0 \\ 0 & 0 & \pi^2/3 \end{pmatrix} \quad ,$$

Note that the values $\pi^2/3$ in this covariance matrix are associated with angles. These values are the variance of a uniform distribution over $[-\pi, \pi]$. In this case the values $\hat{v}, \hat{w}$, required in the calculations, are estimated according to the previous transformation estimate (step k-1):

$$\mathbf{u}' = \hat{T}^{k-1}(\mathbf{u})$$

$$\hat{v} = \mathbf{u}'_x/\mathbf{u}'_z \quad ; \quad \hat{w} = \mathbf{u}'_y/\mathbf{u}'_z \quad .$$

4.4 Optimal Transformation and its Uncertainty

The uncertainty generated from the raw input-data is propagated into uncertainty of the solution (i.e. estimated $3D$ location). Solution uncertainty denotes the belief associated with the estimated location of the object. This uncertainty will be represented by a covariance matrix whose dimension is equal to the degrees of freedom of the transformation (6 for a rigid 3-D transformation). The optimal estimate will be that which minimizes this covariance matrix (or rather minimizes its trace). When representing the transformation as a 6 component vector, the dependence between the estimated rotation and estimated translation is expressed through the entries in the covariance matrix. This dependence is not considered in methods where the process of determining the rotation is separated from the determination of the translation [67, 3, 30, 29].

The representation of the estimation uncertainty as a covariance matrix has the following advantages:

- It is a compact and efficient representation of the information contributed by each feature: Once the data contributed by a feature is fused into the estimation, that feature can be eliminated since it is of no further use. All the information is encoded in the estimation of the transformation and its associated covariance matrix.

- It allows easy parallel or serial fusion of information from different sources and from different types of measured data.

- Acquiring an additional sensed-feature to be matched, in order to refine the estimation, can be done using a statistical goodness of fit test. This test chooses between possible matches under the rigidity constraint defined by the current estimate and thus expedites the searching process in the possible-matching tree.

- The more synthesized features fused, the greater the accuracy of the transformation estimate. The process is such that given a new synthesized feature, we update the previous estimation and decrease the associated uncertainty. The measure of uncertainty allows control over the number of required features; when the uncertainty is sufficiently small the process can be terminated.

4.4.1 The System Definition

1. *The variables to be estimated*:
   The representation of the transformation is composed of two components:

   - The translation component is expressed by the vector t;

     $$t = (t_x, t_y, t_z)^t \quad .$$

   - The rotation component is described by the quaternion $\tilde{\mathbf{q}}$ (see Appendix B) [36]:

     $$\tilde{\mathbf{q}} = (q_0, \mathbf{q}) = (q_0, q_1 i + q_2 j + q_3 k) \quad .$$

   The rotation quaternion satisfies the normality constrains: $\tilde{\mathbf{q}}\tilde{\mathbf{q}}^* = q_0^2 + \|\mathbf{q}\|^2 = 1$, where $\tilde{\mathbf{q}}^*$ is the conjugate of $\tilde{\mathbf{q}}$.

   In practice we represent the rotation component by the vector:

   $$\mathbf{s} \equiv \frac{\mathbf{q}}{q_0}$$

   from which the quaternion $\tilde{\mathbf{q}}$ can be reconstructed:

   $$q_0 = \frac{1}{\sqrt{1 + \mathbf{s}^t\mathbf{s}}} \quad ; \quad \tilde{\mathbf{q}} = (q_0, q_0\mathbf{s}) \quad .$$

   The vector $\mathbf{s}$ is a convenient representation of the rotational component; in addition to being minimal (having 3 parameters) the rotation equation is linear in $\mathbf{s}$ as will be shown later. In order to avoid singularities in the representation when $q_0 = 0$ we always use two different object-centered coordinate systems, simultaneously.

   Considering these two components, the parameter vector to be estimated during the filtering process is:

   $$\mathbf{T} = \begin{pmatrix} \mathbf{s} \\ \mathbf{t} \end{pmatrix} \quad .$$

2. *The observations*:
   A model point is represented by a vector $\mathbf{u}_i = (x, y, z)^t$ in an object centered frame of reference, where the index i denotes the step of the process at which this feature is considered (the same model point can be considered many times when there are

several measurements of this point).
$\mathbf{u}'_i = (x', y', z')^t$ - is the real position of the point $\mathbf{u}_i$ in the viewer centered frame of reference.
$\hat{\mathbf{u}}'_i$ - is the measured position of the point $\mathbf{u}'_i$. This measurement is imprecise and can be represented as:

$$\hat{\mathbf{u}}'_i = \mathbf{u}'_i + \epsilon_i \quad ,$$

where ϵ_i is white noise satisfying:

$$\begin{aligned} E\{\epsilon_i\} &= 0 \\ \mathrm{var}\{\epsilon_i\} &= \Lambda_i \\ \mathrm{cov}\{\epsilon_i, \epsilon_j^t\} &= 0 \quad \forall i \neq j \quad . \end{aligned}$$

3. *The measurement model*:
A mathematical relationship between the measured vector and the estimated vector is expressed, for each feature i, by a non linear quaternion equation:

$$\tilde{\mathbf{u}}'_i = \tilde{\mathbf{q}}\tilde{\mathbf{u}}_i\tilde{\mathbf{q}}^* + \tilde{\mathbf{t}} \quad , \tag{4.1}$$

where $\tilde{\mathbf{u}}_i, \tilde{\mathbf{u}}'_i, \tilde{\mathbf{t}}$ are quaternions associated with the vectors $\mathbf{u}_i, \mathbf{u}'_i, \mathbf{t}$ respectively. Given that $\tilde{\mathbf{q}}\tilde{\mathbf{q}}^* = 1$, multiplying Equation (4.1) by $\tilde{\mathbf{q}}$ yields:

$$\tilde{\mathbf{u}}'_i\tilde{\mathbf{q}} = \tilde{\mathbf{q}}\tilde{\mathbf{u}}_i + \tilde{\mathbf{t}}\tilde{\mathbf{q}} \quad .$$

Isolating the vector component of this quaternion equation and dividing by q_0 we get the matrix equation:

$$h_i(\mathbf{u}_i, \mathbf{u}'_i, \mathbf{T}) \equiv <\mathbf{u}'_i + \mathbf{u}_i> \mathbf{s} + (\mathbf{u}'_i - \mathbf{u}_i) - (I_3 - <\mathbf{s}>)\mathbf{t} = \mathbf{0} \quad , \tag{4.2}$$

where $\mathbf{s} \equiv \frac{\mathbf{q}}{q_0}$ as previously defined, I_3 is the 3×3 identity matrix and $<\cdot>$ denotes the matrix form of a cross product, i.e:

$$<\mathbf{v}> = \begin{pmatrix} 0 & -v_z & v_y \\ v_z & 0 & -v_x \\ -v_y & v_x & 0 \end{pmatrix} \quad ; \quad <\mathbf{v}>\mathbf{u} = -<\mathbf{u}>\mathbf{v} = \mathbf{v} \times \mathbf{u} \quad .$$

Notice that according to the definition of the measurement noise, we assume no correlation between the different measurement noise ($\mathrm{cov}\{\epsilon_i, \epsilon_j\} = 0 \; \forall i \neq j$). This assumption is not always valid. When there is correlation between several measurements, we may consider these measurements as a single measurement by grouping the measurement values into a single vector and by combining their corresponding equations into a single vector equation.

4.4.2 The Estimation Control

The estimation process is composed of an incremental process, for which at each step $k-1$, there exists an estimate $\hat{\mathbf{T}}^{k-1} = \begin{pmatrix} \hat{\mathbf{s}}^{k-1} \\ \hat{\mathbf{t}}^{k-1} \end{pmatrix}$ of the transformation $\mathbf{T}$ and a covariance matrix Σ^{k-1} which represents the "quality" of the estimate $\hat{\mathbf{T}}^{k-1}$. Given a new match $(\mathbf{u}_k, \hat{\mathbf{u}}'_k)$ the current estimate is updated to be $\hat{\mathbf{T}}^k$ with the associated uncertainty Σ^k. The accuracy of the estimate increases, as additional matches are fused, i.e. $\Sigma^k \leq \Sigma^{k-1}$ ($\Sigma^{k-1} - \Sigma^k$ is nonnegative definite). The process terminates as soon as the uncertainty satisfies our criterion for accuracy or no additional match can be supplied [32].

Fusing the information from a match with the old estimate is performed using the *Extended Kalman filter* (E.K.F.) process which is a generalization of the *Kalman Filter* (K.F.) to non-linear systems [43, 55]. The K.F. equations are given in Section 3.2. The transition from step $k-1$ to step k is performed by using a linear approximation of the equation h_k defined in (4.2). Linearization of h_k is obtained by taking the first order Taylor expansion around $(\hat{\mathbf{T}}^{k-1}, \hat{\mathbf{u}}'_k)$:

$$h_k(\mathbf{u}_k, \mathbf{u}'_k, \mathbf{T}) = 0 \approx h_k(\mathbf{u}_k, \hat{\mathbf{u}}'_k, \hat{\mathbf{T}}^{k-1}) + \frac{\partial h_k}{\partial \mathbf{u}'_k}(\mathbf{u}'_k - \hat{\mathbf{u}}'_k) + \frac{\partial h_k}{\partial \mathbf{T}}(\mathbf{T} - \hat{\mathbf{T}}^{k-1}) \tag{4.3}$$

which yields:

$$<\hat{\mathbf{s}}^{k-1}>\hat{\mathbf{t}}^{k-1} + \mathbf{u}_k - \hat{\mathbf{u}}'_k \approx [I_3 - <\hat{\mathbf{s}}^{k-1}>](\mathbf{u}'_k - \hat{\mathbf{u}}'_k) + [<\hat{\mathbf{u}}'_k + \mathbf{u}_k - \hat{\mathbf{t}}^{k-1}>, (<\hat{\mathbf{s}}^{k-1}> - I_3)]\mathbf{T} \quad . \tag{4.4}$$

Equation (4.4) can be rewritten as a linear equation:

$$\mathbf{z}_k = H_k \mathbf{T} + \eta_k \quad , \tag{4.5}$$

where

$$\begin{aligned} \mathbf{z}_k &= <\hat{\mathbf{s}}^{k-1}>\hat{\mathbf{t}}^{k-1} + \mathbf{u}_k - \hat{\mathbf{u}}'_k \\ H_k &= [<\hat{\mathbf{u}}'_k + \mathbf{u}_k - \hat{\mathbf{t}}^{k-1}>, (<\hat{\mathbf{s}}^{k-1}> - I_3)] \\ \eta_k &= [I_3 - <\hat{\mathbf{s}}^{k-1}>](\mathbf{u}'_k - \hat{\mathbf{u}}'_k) \quad . \end{aligned} \tag{4.6}$$

$\mathbf{z}_k$ represents the new "measurement", H_k is the matrix denoting a linear connection between the "measurement" and the actual transformation $\mathbf{T}$. Both $\mathbf{z}_k$ and H_k can be derived from $\hat{\mathbf{u}}'_k$, $\mathbf{u}_k$, $\hat{\mathbf{T}}^{k-1}$. The term η_k - depicts the noise in the "measurement" $\mathbf{z}_k$ and satisfies:

$$\begin{aligned} E\{\eta_k\} &= 0 \\ \mathrm{var}\{\eta_k\} &= [I_3 - <\hat{\mathbf{s}}^{k-1}>]\Lambda_k[I_3 - <\hat{\mathbf{s}}^{k-1}>]^t = W_k \\ \mathrm{cov}\{\eta_k, \eta_\ell^t\} &= 0 \quad \forall\ k \neq \ell \quad . \end{aligned}$$

The K.F. procedure refines the estimate $\hat{\mathbf{T}}^{k-1}$ by fusing an additional match $(\mathbf{u}_k, \hat{\mathbf{u}}'_k)$. The process is of the form:

$$\hat{\mathbf{T}}^k = f(\hat{\mathbf{T}}^{k-1}, \Sigma^{k-1}, \mathbf{u}_k, \hat{\mathbf{u}}'_k, \Lambda_k, h_k)$$

obtaining from the K.F. updating equations. Thus, at each stage k, there is no need of retaining any of the previously considered matches. Only the current estimate $\hat{\mathbf{T}}^{k-1}$ and its associated uncertainty Σ^{k-1} need be retained.

Further accuracy can be achieved in the approximated equation (4.5) by adding the higher order terms derived from the Taylor expansion. Only the first and the second order terms do not vanish when expanding h_k. The first order terms have already been taken into account when Equation (4.5) was derived, calculating the higher order terms which do not vanish gives:

$$\begin{aligned} (\mathbf{u}'_k - \hat{\mathbf{u}}'_k)^t \frac{\partial^2 h_k}{\partial \mathbf{u}'_k \partial \mathbf{T}}(\mathbf{T} - \hat{\mathbf{T}}^{k-1}) + \frac{1}{2}(\mathbf{T} - \hat{\mathbf{T}}^{k-1})^t \frac{\partial^2 h_k}{\partial \mathbf{T}^2}(\mathbf{T} - \hat{\mathbf{T}}^{k-1}) = \\ (\mathbf{u}'_k - \hat{\mathbf{u}}'_k) \times (\mathbf{s} - \hat{\mathbf{s}}^{k-1}) + (\mathbf{s} - \hat{\mathbf{s}}^{k-1}) \times (\mathbf{t} - \hat{\mathbf{t}}^{k-1}) \quad . \end{aligned} \tag{4.7}$$

We can calculate the expectation values of these two terms: The left term is zero since $(\mathbf{u}'_k - \hat{\mathbf{u}}'_k)$ is a white process and the second term can be reconstructed from the covariance matrix Σ^{k-1}. The higher order expectation values are subtracted from the "measurement" $\mathbf{z}_k$ in Equation 4.5.

Implementing the K.F. on Equation (4.5) yields an unbiased estimate of $\mathbf{T}$ which is optimal in the linear minimal variance criterion [2], i.e. $\hat{\mathbf{T}}^k$ minimizes

$$C = (\mathbf{T} - \hat{\mathbf{T}}^0)(\Sigma^0)^{-1}(\mathbf{T} - \hat{\mathbf{T}}^0)^t + \sum_{i=1}^{k}(\mathbf{z}_i - H_i\mathbf{T})W_i^{-1}(\mathbf{z}_i - H_i\mathbf{T})^t \quad ,$$

In the case where the measurement noise ε is a Gaussian process the K.F. gives an estimate which is also optimal in the sense of maximum-likelihood criterion. In this case, the estimate coincides with the conditional expectation:

$$\hat{\mathbf{T}}^k = E\{\mathbf{T}|\{\hat{\mathbf{u}}_i'\}_{i=1\ldots k}\} \quad .$$

4.5 Computational Aspects of the Method

4.5.1 Stability

There are several important computational problems concerning the K.F and related to its instability. It is well known that the K.F. process is not stable [55, page 399] since it involves matrix inversions which are not necessarily well-posed. This is particularly true in our case where the measurements are obtained from projections and the eigenvalues of Λ_k are widely distributed. This increases the condition-number of the inverted matrix, $M = H_k\Sigma^{k-1}H_k + \Lambda_k$ making it ill-posed. Furthermore, the imprecision due to linearization of the non-linear equations, may influence the system to give non feasible solutions. Imprecise linear approximations often occur at the beginning of the process when linearization is about a point which is not sufficiently close to the true solution. In general, most of the instability problems arise at the initial stages of the process (according to our experience - during the first 5-8 2D-3D matches). The later stages of the process are stable and converge nicely. Thus, during the initial stages of the process it is important to perform careful and exact evaluations even at the cost of additional computations. There are several strategies that can contribute to the stability of the process:

1. Stable computation:
   In order to avoid computational instabilities several variations of the K.F. are used during the process according to the quality of information accumulated about the

estimated vector. In general, in order to reduce the problem of representing a wide range of values in a limited wordlength and to improve the condition number of M it is preferable to use the *square root filter* [55]. With this implementation it is assured that Σ will be a non-negative matrix and we double the effective precision of its values (due to the limited range of values of the square root matrix). It can be shown that the computation using the square root filter is numerically stable (see [55, page 399]).

The *inverse square root filter* [55] uses only the Σ^{-1} matrix and not Σ. Thus, in the early stages of the process, when Σ^{-1} is not of full rank, the inverse square root filter should be used. During these stages the state vector is represented as:

$$\hat{\mathbf{y}}_i = \Sigma_i^{-1}\hat{\mathbf{s}}_i \quad .$$

As additional features are considered the matrix Σ^{-1} becomes well conditioned and a more reliable estimate of the transformation can be performed. At this stage we transform the above representation to the standard representation of $\mathbf{s}$ by:

$$\hat{\mathbf{s}}_i = \Sigma_i \hat{\mathbf{y}}_i$$

and continue the process using the square root filter.

2. Fusing information from several measurements:
   Stability of the process can be increased in its early stages by considering several measurements at once. This is done by grouping equations of a number of measurements into a single vector equation. The number of features to be grouped together should be equal or greater to the number of correspondences that constrain the matching to have a finite set of solutions. This assures that Σ^{-1} is full ranked at every step. On the other hand, it is important not to use too many measurements at a single step, as this will deteriorate the linear approximation of the non-linear equations used by the E.K.F. Therefore, the optimal number of measurements to be grouped should be that which is minimal, yet still assures Σ^{-1} is full ranked, i.e., three points for $3D$ or perspective projection and four points for orthographic projection.

3. Choosing a reasonable initial estimate:
   The K.F. process assumes an initial estimate $\hat{\mathbf{T}}^0$ with uncertainty Σ^0. This in-

formation is not necessarily available, and must be extracted from the data. The importance of choosing a reasonable estimate is due to the linearization of the equations $\{h_i\}$. Since the linear approximation of h_1 is performed around $\hat{\mathbf{T}}^0$, an unreasonable initial estimate may influence the approximation so that the process will not converge to the correct solution.

Following many simulations, it was found that even unreasonable initial estimations do converge to the expected solution using local iterative K.F. [43, page 349] at the initial stages of the process. The use of local iterations is effective in limiting the negative influence of the linear approximation and induces quick convergence (up to 7 iterations in our algorithm) to a reasonable estimate. For the same reason, it is also recommended to fuse the "good" measurements (measurements with low uncertainty) at the early stages of the process. At further stages of the process the use of local iterations does not significantly improve the transformation estimate and can be discontinued.

In any case, it is possible to give a crude estimate using simple techniques (for example: [3, 67, 49]) utilizing a small number of matched features. The initial uncertainty Σ^0 is chosen to be a diagonal matrix with very large values, so that the influence of the initial estimate $\hat{\mathbf{T}}^0$ on the final solution is marginal.

4. Eliminating outlier measurements:
   As noted in Section 3.6, modeling the measurement noise as a Gaussian process in the K.F. equations, is not always exact due to the several possible outlier measurements which may impair the solution. The matching strategy that will be described in Section 4.6.1 can contribute to the extraction of these outlier measurements.

4.5.2 Convergence

The convergence of the estimate to the true state vector can be evaluated by studying the qualitative behavior of the covariance matrix Σ. The evolution of Σ during the process is given by (See Equation 3.4):

$$(\Sigma^{t+1})^{-1} = H_{t+1}^t \Lambda_{t+1}^{-1} H_{t+1} + (\Sigma^t)^{-1} \quad .$$

Under the assumption that H and Λ are constant along time, we have:

$$\frac{\partial \Sigma^{-1}(t)}{\partial t} \approx (\Sigma^{t+1})^{-1} - (\Sigma^{t})^{-1} = H^t \Lambda^{-1} H$$

so that:

$$\Sigma(t) \sim \frac{(H^t \Lambda^{-1} H)^{-1}}{t}$$

i.e. the convergence of Σ is at a rate of t^{-1} and thus the squared deviation $\|\mathbf{T} - \hat{\mathbf{T}}^t\|^2$ also converges like t^{-1}.

4.5.3 Complexity

In our case, the measurement $\mathbf{z}$ and the state vector $\mathbf{T}$ are of dimensions three and six respectively, thus, each application of the K.F. fuser takes constant time. As previously described (Section 3.5), the local iterations are equivalent to solving a non-linear system of equations using Newton iterations which is known for its rapid convergence. In our case, up to 7 iterations were required at the first stages of the process and at further stages local iterations were unnecessary. Therefore, it can be said that the number of local iterations is limited and the complexity for estimating pose, given k measurements, is $O(k)$. In our application calculating the pose using 150 measurements required 1-2 seconds computing time on a Sun 4 workstation.

4.6 Additional Advantages

4.6.1 The Correspondence Process

Determining the matching between the measurements and the model points (i.e. determining the measurement interpretation) is a difficult problem in itself. Up till now, the matching was assumed to be given. This assumption is reasonable for problems such as hypothesis verification where the pose estimation of an object serves to confirm or reject an interpretation hypothesis (see [42, 69, 12])). Likewise, in problems of motion tracking, it is often assumed that the matching is known when the object location changes little

from frame to frame since easy tracking of features can be applied. However the assumption that the matching is known, is not acceptable in all pose estimation problems, particularly when dealing with recognition.

One of the methods suggested to solve the matching problem [22, 20, 6] is to evaluate the pose and the matching simultaneously; i.e. during the iterative process, the pose of the object is estimated from a partial interpretation and this pose estimate is used to eliminate irrelevant interpretations at the next interpretation stage etc. This method is easily applied to the estimation process we suggested. The use of the K.F. process enables us to obtain reasonable matches during the estimation process [22, 20]; at each step i, use the current estimate $\hat{\mathbf{T}}^i$ and its corresponding confidence Σ^i to select "good" matches. The selection is done using goodness of fit tests:
Given a model-feature $\mathbf{u}_k$, let a candidate for a match be the measurement $(\hat{\mathbf{u}}'_k, \Lambda_k)$. According to this hypothesis, $\hat{h}_k = h_k(\mathbf{u}_k, \hat{\mathbf{u}}'_k, \hat{\mathbf{T}}^{k-1})$ (Equation 4.3) is an independent random variable with a normal distribution which satisfies:

$$E\{\hat{h}_k\} = 0$$

$$\mathrm{var}\{\hat{h}_k\} = (\frac{\partial h_k}{\partial \mathbf{u}'_k})\Lambda_k(\frac{\partial h_k}{\partial \mathbf{u}'_k})^t + (\frac{\partial h_k}{\partial \mathbf{T}})\Sigma^{k-1}(\frac{\partial h_k}{\partial \mathbf{T}})^t \equiv S_k \quad .$$

The "goodness" of fit between $\mathbf{u}_k$ and $\hat{\mathbf{u}}'_k$ is then given by the Mahalanobis distance:

$$d(\mathbf{u}_k, \hat{\mathbf{u}}'_k) = (\hat{h}_k)S_k^{-1}(\hat{h}_k)^t = g(\hat{\mathbf{T}}^k, \Sigma^k) \quad ,$$

where g has a χ^2 distribution with $rank(S_k)$ degrees of freedom. The probability that the match is correct is inverse to g. If g is greater than a predefined small value, the candidate match is rejected and another candidate is examined. As the process proceeds, the uncertainty Σ decreases and then S^{-1} increases, lowering the number of acceptable matches.

An example for the matching strategy is given in Figure 4.3. In this figure $\mathbf{u}'_k$ denotes the estimated position of $\mathbf{u}_k$ based on the transformation $\hat{\mathbf{T}}^{k-1}$. The ellipse surrounding $\mathbf{u}'_k$ represents the uncertainty of this estimated position due to the uncertainty of $\hat{\mathbf{T}}^{k-1}$. Two measured features $\hat{\mathbf{u}}'_{(k)_i}$ and $\hat{\mathbf{u}}'_{(k)_j}$ are candidates for a possible match and their associated uncertainties are represented by the surrounding ellipses. According to the situation depicted in this figure it is clear that $(\mathbf{u}_k, \hat{\mathbf{u}}'_{(k)_j})$ is a better match than $(\mathbf{u}_k, \hat{\mathbf{u}}'_{(k)_i})$

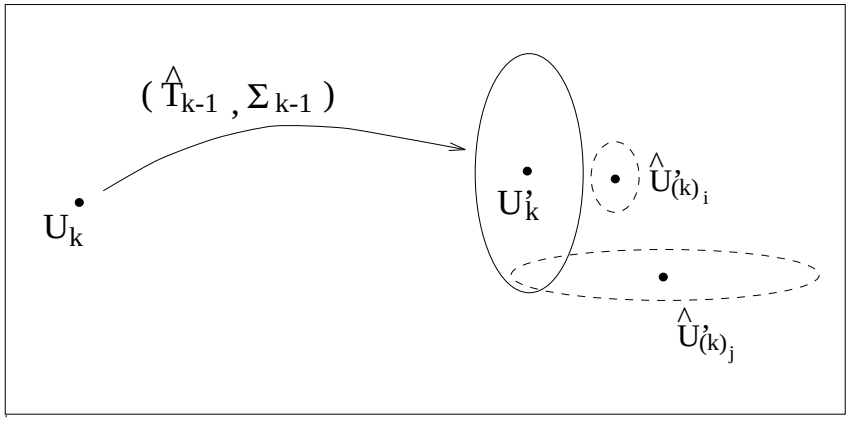

Figure 4.3: $\mathbf{u}'_k$ is the expected position of the measured point. $\hat{\mathbf{u}}'_{(k)_j}$ and $\hat{\mathbf{u}}'_{(k)_i}$ are candidates to be matched with it. $\hat{\mathbf{u}}'_{(k)_j}$ is a better match than $\hat{\mathbf{u}}'_{(k)_i}$ even though the latter is closer to $\mathbf{u}'_k$, with respect to Euclidean distance.

even though the latter match is of a shorter Euclidean distance. From this example it can be seen that the uncertainties associated with the measured features and the uncertainty associated with the transformation estimate supply important information to the matching process.

In many cases a good match for the model point $\mathbf{u}_k$ can not be found due to occlusion or inability to obtain information about certain interest points in the image. In these cases we synthesize an artificial measurement for the model point and associate it with an infinite uncertainty so that its influence on the rest of the process will be minimal.

4.6.2 Parallelization of the Process

Assume that the E.K.F. process was performed on two separate channels a and b, using n matches in each channel:

$$\begin{aligned} \{\hat{\mathbf{u}}'_i\}_{i=1\ldots n} &\xrightarrow{E.K.F.} \hat{\mathbf{T}}_a, \Sigma_a \\ \{\hat{\mathbf{u}}'_i\}_{i=n+1\ldots 2n} &\xrightarrow{E.K.F.} \hat{\mathbf{T}}_b, \Sigma_b \end{aligned}$$

Optimal fusion of the $2n$ matches will be performed as follows:

$$\hat{\mathbf{T}}_{ab} = \hat{\mathbf{T}}_a + K(\hat{\mathbf{T}}_b - \hat{\mathbf{T}}_a)$$

$$\Sigma_{ab} = (I - K)\Sigma_a$$

where

$$K = \Sigma_a(\Sigma_a + \Sigma_b)^{-1} \quad .$$

This result can be obtained easily from the K.F. equations if we interpret $(\hat{\mathbf{T}}_a, \Sigma_a)$ as an *a-priori* estimation of $\mathbf{T}$ and considered $\hat{\mathbf{T}}_b$ as a "new measurement" with associated covariance matrix Σ_b which satisfies Equation (4.5) such that [1]:

$$\mathbf{z} = \hat{\mathbf{T}} \quad , \quad H = I \quad \text{and} \quad \text{var}\{\eta\} = \Sigma_b \quad .$$

An extension of this method can fuse estimates obtained from a greater number of channels (requiring $\log m$ steps for m channels). It is thus possible to decompose the K.F. process into several channels and then fuse the obtained estimates into a single optimal solution. This framework can also be used to fuse information from several kinds of measured primitives, e.g. lines and planes, where each channel is dedicated to one kind of primitive.

Note that in this process, the initial estimate $\hat{\mathbf{T}}^0$ and the initial uncertainty Σ^0, are considered in each of the channels. Thus it must be ensured that Σ^0 is very large so that the accumulated error due to repeated consideration of $\hat{\mathbf{T}}^0$, can be neglected. Simulation results of the parallel process show convergence equivalent to the serial process as will be elaborated later on.

4.6.3 Uncertain Model Features

In this thesis we assume an exact model with no uncertainties, but a straightforward extension to the proposed method will allow us to include non-exact model features that may arise from imprecise modeling of 3D objects (for example when modeling faces or other semi-elastic objects). In this case linearization of $h_k(\mathbf{u}_k, \mathbf{u}'_k, \mathbf{T})$ (Eq 4.3) is performed around $(\hat{\mathbf{u}}_k, \hat{\mathbf{u}}'_k, \hat{T}^{k-1})$ where $\hat{\mathbf{u}}_k$ is the estimate of the model point with uncertainty

$$\text{var}\{\hat{\mathbf{u}}_k\} = \zeta_k$$

[1] The result does not dependent on which of the estimates, $\mathbf{T}_a$ or $\mathbf{T}_b$, is chosen to be the *a-priori* estimation and which is chosen to be the measurement.

The linear approximation yields:

$$h_k(\mathbf{u}_k, \mathbf{u}'_k, \mathbf{T}) = 0 \approx h_k(\hat{\mathbf{u}}_k, \hat{\mathbf{u}}'_k, \hat{\mathbf{T}}^{k-1}) + \frac{\partial h_k}{\partial \mathbf{u}'_k}(\mathbf{u}'_k - \hat{\mathbf{u}}'_k) + \frac{\partial h_k}{\partial \mathbf{T}}(\mathbf{T} - \hat{\mathbf{T}}^{k-1}) + \frac{\partial h_k}{\partial \mathbf{u}_k}(\mathbf{u}_k - \hat{\mathbf{u}}_k) \tag{4.8}$$

which yields a linear equation as given in Eq. 4.5 but where

$$\eta_k = [I_3 - <\hat{\mathbf{s}}^{k-1}>](\mathbf{u}'_k - \hat{\mathbf{u}}'_k) - [I_3 + <\hat{\mathbf{s}}^{k-1}>](\mathbf{u}_k - \hat{\mathbf{u}}_k)$$

and $\mathbf{z}_k, H_k$ are the same as in Eq. 4.6.

4.7 Simulation Results

We tested our paradigm by simulating a model as a collection of points. Points of the model were chosen by randomly sampling the space $[0, 100]^3$. The model points were transformed by a transformation $\mathbf{T}$ composed of a rotation $\mathbf{s}$ and a translation $\mathbf{t}$. The measurements of the transformed points were contaminated by white Gaussian noise. The algorithm estimates the transformation $\mathbf{T}$ using four types of measurements:

- $1D$ measurements of distances from the observer.
- $2D$ measurements of the perspective projection.
- $2D$ measurements of the orthographic projection.
- $3D$ measurements.

The algorithm assumes the correspondence between the points of the model and the measured points is known.

A sample of results obtained by the simulations are presented in graphs 4.4-4.17. Graphs 4.4 and 4.5 show the convergence of the estimates of the rotation $\hat{\mathbf{s}}$ and the translation $\hat{\mathbf{t}}$ as a function of the number of measurements (matched points). The vertical ordinate represents the normalized error of the estimate:

$$t_i^{error} = \frac{\|\hat{\mathbf{t}}_i - \mathbf{t}\|}{\|\mathbf{t}\|} \qquad \text{in Graph 4.4}$$

$$s_i^{error} = \frac{\|\hat{\mathbf{s}}_i - \mathbf{s}\|}{\|\mathbf{s}\|} \qquad \text{in Graph 4.5} \quad .$$

In each of these graphs, three cases are displayed; noise with s.t.d. of 10, 5 and 0. It can be seen that the convergence rate of the estimate is quite fast in the initial stages of the process and corresponds to a decay of rate t^{-1} as expected (see Section 4.5.2). The convergence rate depends on the amount of noise in the measurements, however, in all cases, following quick initial convergence further improvement of the estimate requires an increasing number of measurements. When measurements are noise free, convergence to the correct solution is immediate (3-6 matches). This is true also for measurements in $2D$ as can be seen in graphs 4.7,4.8,4.10 and 4.11 (5-20 matches).

Graph 4.6 depicts the trace of the covariance matrix corresponding to the estimate $\hat{\mathbf{T}}$: $E\{\|\mathbf{T}-\hat{\mathbf{T}}\|^2\}$ in comparison with the squared deviation of $\hat{\mathbf{T}}$ from the true transformation: $\|\mathbf{T}-\hat{\mathbf{T}}\|^2$. The graph shows a high correlation between the estimate quality and the certainty its covariance matrix describes. This behavior indicates that the algorithm indeed exploits the supplied information.

Graphs 4.7-4.9 and 4.10-4.12 represent simulations, similar to those described above, for the cases of orthographic and perspective projections, respectively. Here too, convergence is quick and stable. Note, that in the case of orthographic projection, the z-component of the transformation cannot be estimated and remains as a free parameter throughout the process. Thus, the graphs corresponding to this case leave the z-component out. In the case of perspective projection, noise is added to the measurements in the image plane. Gaussian noise with s.t.d of 0.1, as presented in the graphs (4.10-4.12), is equal in our camera configuration to 2% of the total size of the body as observed in the image plane. This corresponds to about four pixels of error in an object such as in Figure 4.18.

Graphs 4.13-4.15 represent simulation for integrated measurements. $1D$, perspective projection, orthographic projection and $3D$ data measurements are fused in a interleaved series, one measurement type at time. The graphs depict two cases: One case where the s.t.d for perspective, orthographic, $3D$ and $1D$ measurements are 0.2, 10.0, 5.0 and 10.0 respectively, and the other where the s.t.d are 0.3, 5.0, 5.0 and 2.0.

Graphs 4.16,4.17 show some simulation results for the parallel process. $3D$ uncertain measurements, contaminated with Gaussian noise of s.t.d of 10, are divided into 16 sets of measurements, 12 measured points each. In the first step the estimated transformations

were calculated in parallel for each of the 16 sets. In the consequent steps fusion was performed in a binary tree fashion, were the 16 sets were considered as the leaves of the tree. Each node of the tree represents the result obtained from fusion of the data of its ancestral leaves. In our case the tree has 4 levels ($\log 16$).

Graphs 4.16,4.17 compare the convergence of the serial process to that of the parallel process, for the same set of measurements. These graphs show the convergence of the normalized deviation of the estimated translation and the estimated rotation, respectively. It can be seen that the parallel process converges to the same solution of the serial process. Furthermore, there is good correlation between convergence of the serial process and the average of the nodes at each level of the tree, in the parallel process.

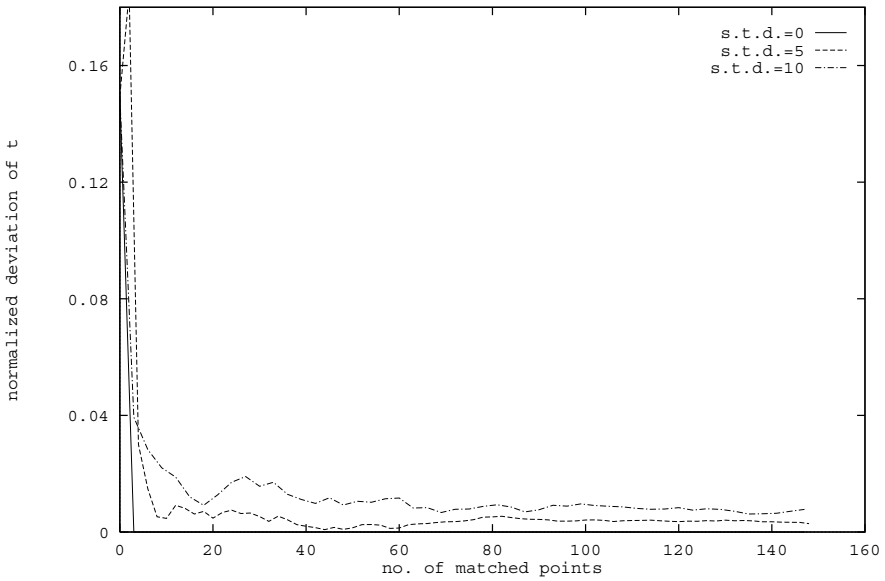

Figure 4.4: $3D$ measurements: Normalized deviation of the translation estimate $\hat{\mathbf{t}}$.

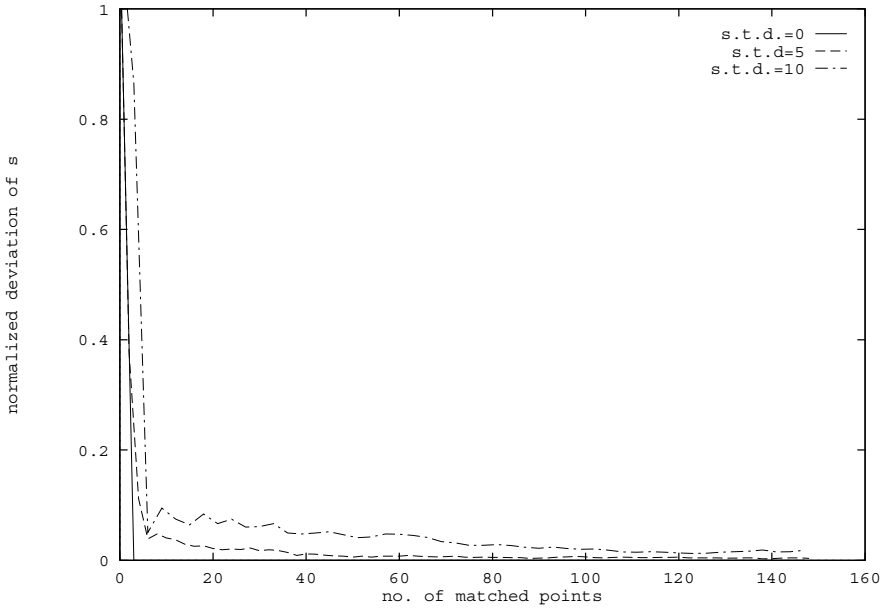

Figure 4.5: $3D$ measurements: Normalized deviation of the rotation estimate $\hat{\mathbf{s}}$.

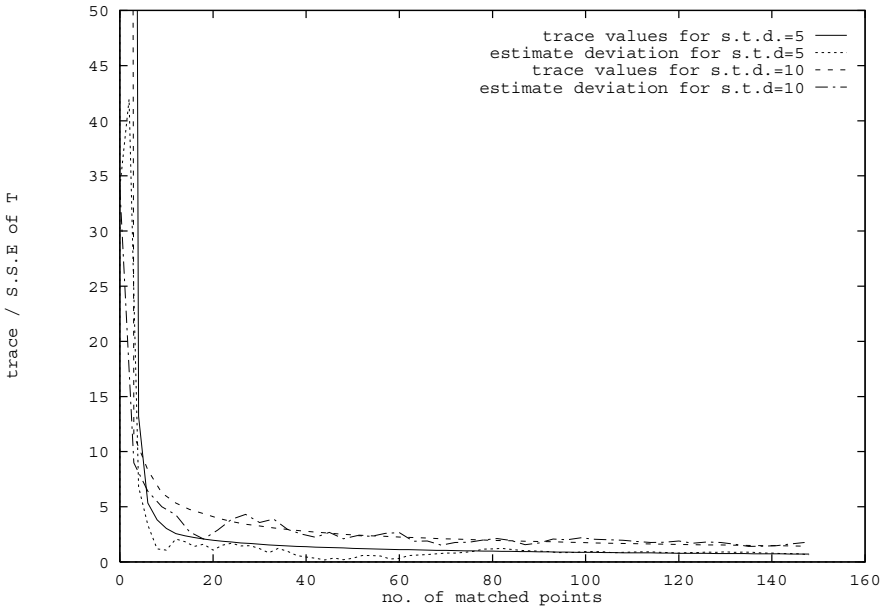

Figure 4.6: $3D$ measurements: Comparison between the trace of Σ and the squared deviation of the estimate from the real transformation.

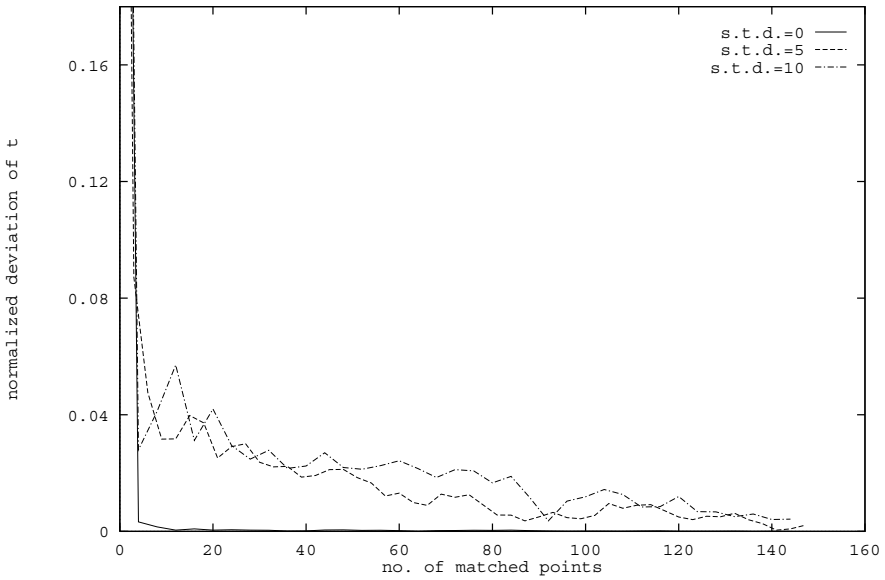

Figure 4.7: Orthographic projection: Normalized deviation of the translation estimate $\hat{\mathbf{t}}$.

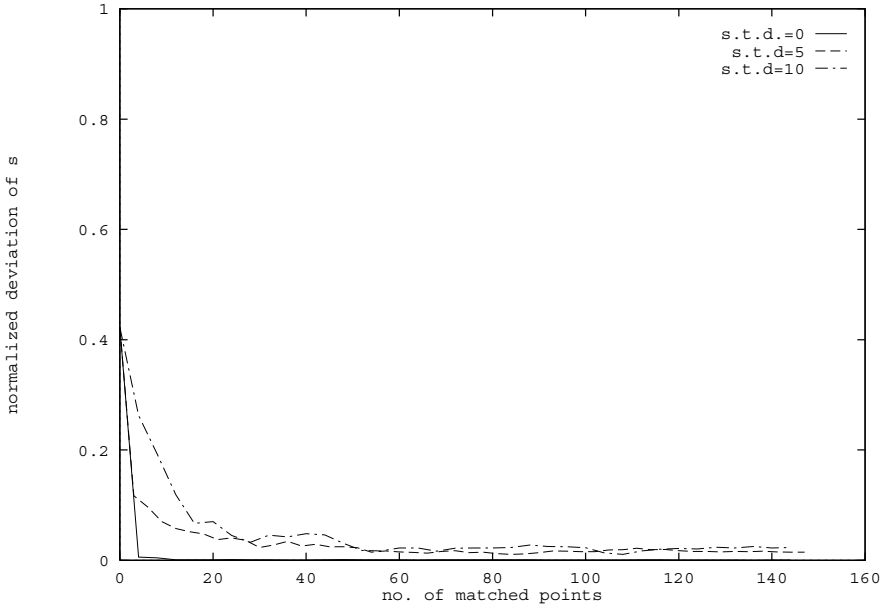

Figure 4.8: Orthographic projection: Normalized deviation of the rotation estimate $\hat{\mathbf{s}}$.

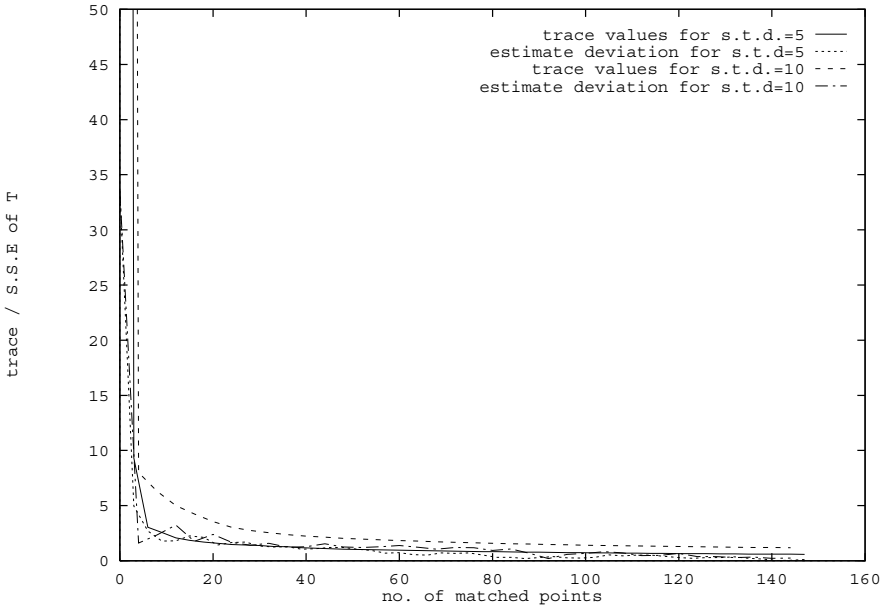

Figure 4.9: Orthographic projection: Comparison between the trace of Σ and the squared deviation of the estimate from the real transformation.

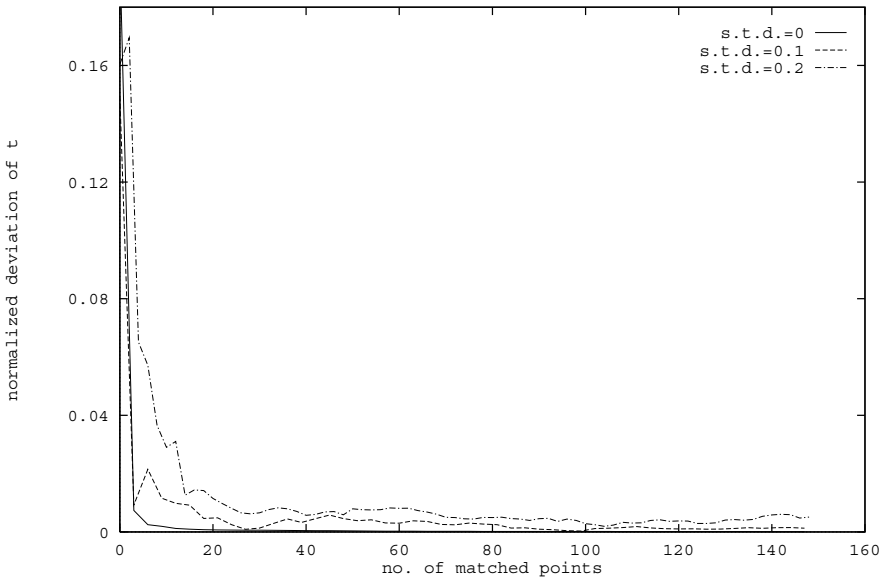

Figure 4.10: Perspective projection: Normalized deviation of the translation estimate $\hat{\mathbf{t}}$.

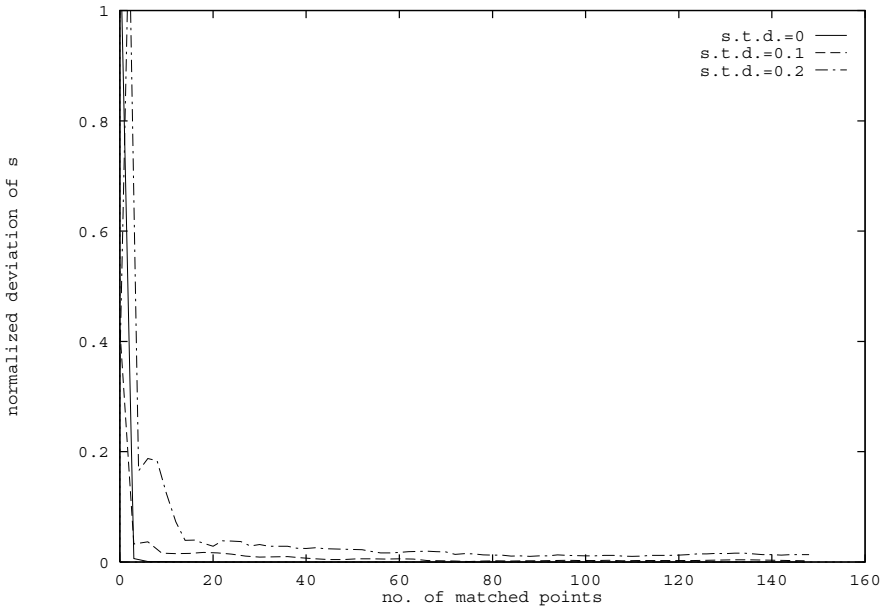

Figure 4.11: Perspective projection: Normalized deviation of the rotation estimate $\hat{\mathbf{s}}$.

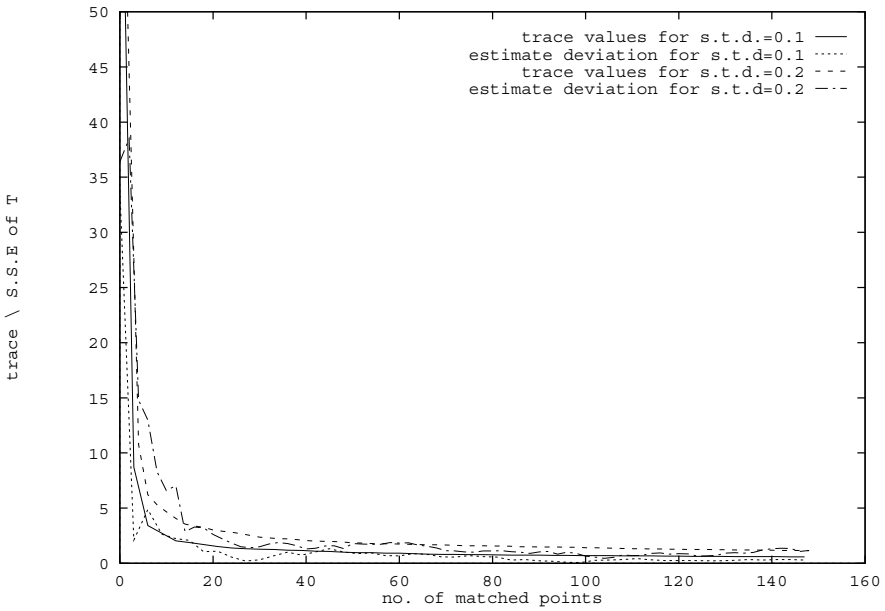

Figure 4.12: Perspective projection: Comparison between the trace of Σ and the squared deviation of the estimate from the real transformation.

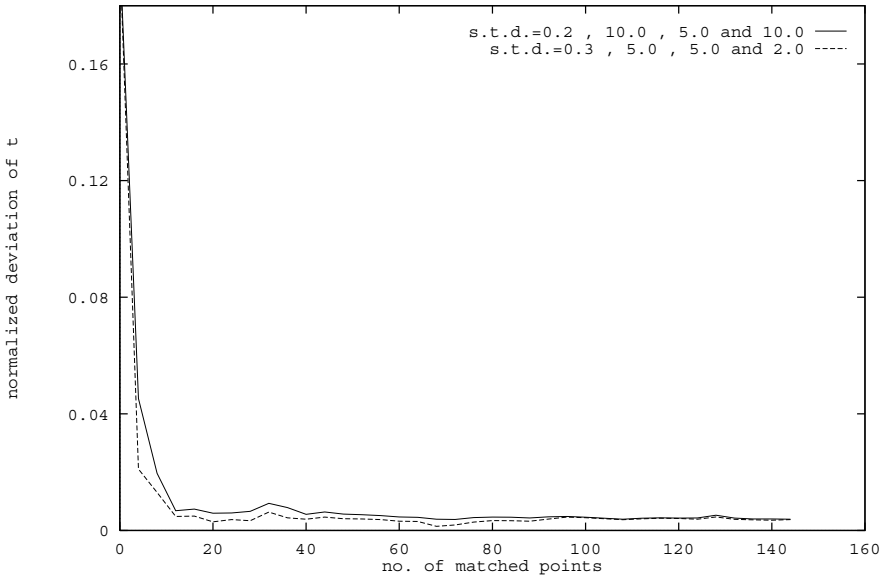

Figure 4.13: Integrated case: Normalized deviation of the translation estimate $\hat{\mathbf{t}}$. The s.t.d. values denoted in the graph are for perspective, orthographic, $3D$ and $1D$ measurements respectively.

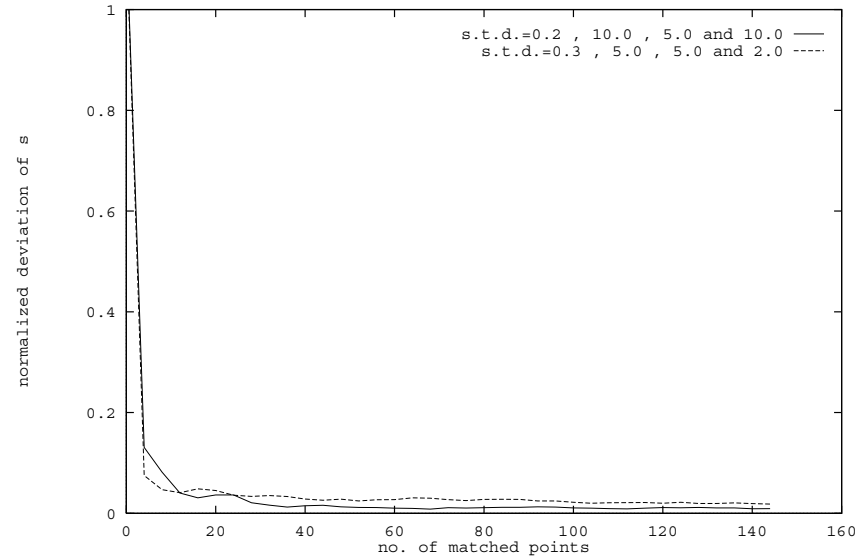

Figure 4.14: Integrated case: Normalized deviation of the rotation estimate $\hat{\mathbf{s}}$. The s.t.d. values denoted in the graph are for perspective, orthographic, $3D$ and $1D$ measurements respectively.

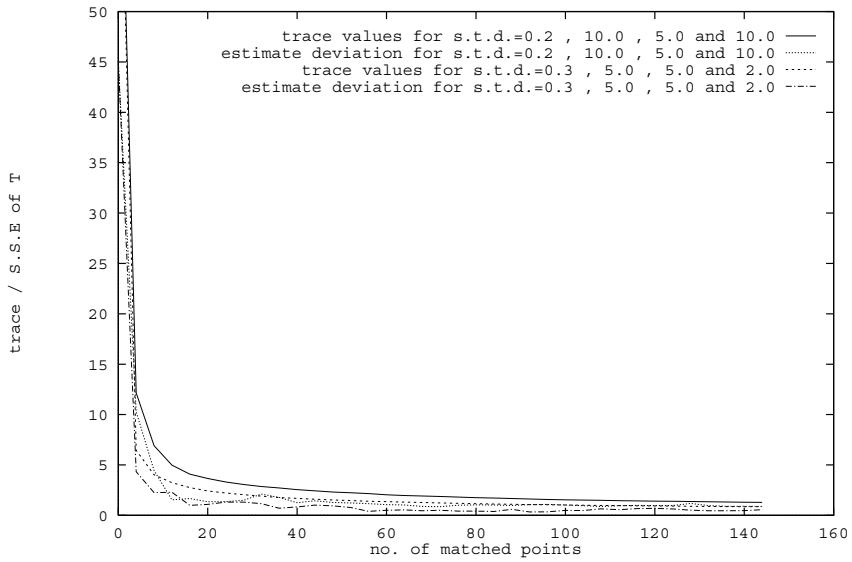

Figure 4.15: Integrated case: Comparison between the trace of Σ and the squared deviation of the estimate from the real transformation. The s.t.d. values denoted in the graph are for perspective, orthographic, $3D$ and $1D$ measurements respectively.

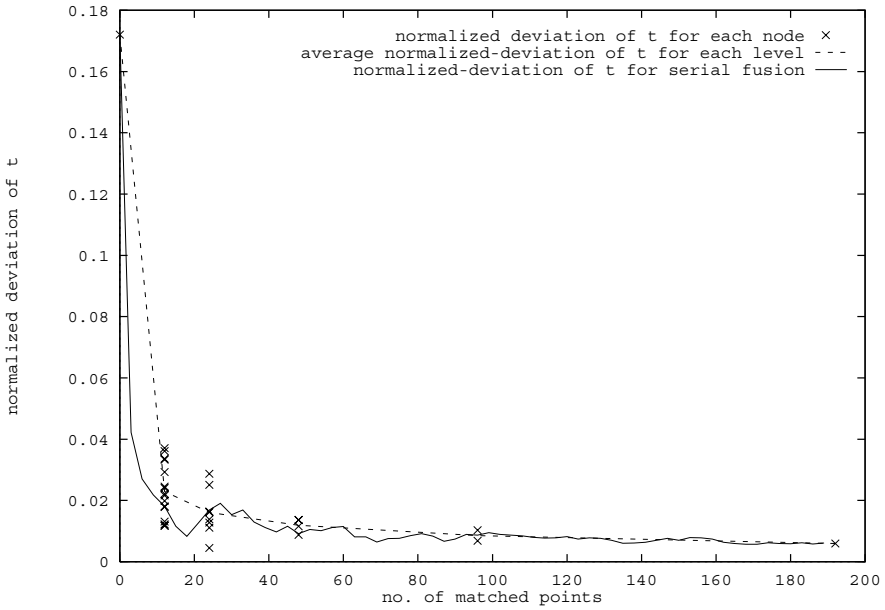

Figure 4.16: Simulation for the parallel process: Normalized deviation of the translation estimate $\hat{\mathbf{t}}$. The measurements are in 3D and the s.t.d. of the contaminated error is 10.

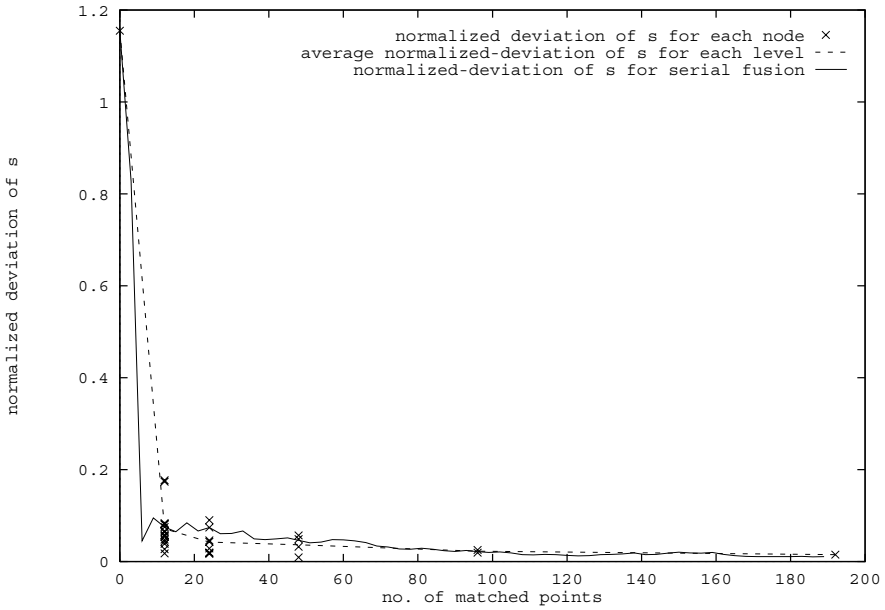

Figure 4.17: Simulation for the parallel process: Normalized deviation of the rotation estimate $\hat{\mathbf{s}}$. The measurements are in 3D and the s.t.d. of the contaminated error is 10.

4.8 Results on Real Images

Our algorithm was applied to measurements taken from $2D$ images of an object (under perspective projection). In the following example we took images of the object at four different positions. In all images the object was placed on a planar table (see the 4 pictures in Figure 4.18) and the real transformation between every two positions was measured (i.e. translation distance and angle of rotation). The algorithm was applied to each of the given images. The measurements consisted of 35 feature points; 29 features were $2D$ measurements taken from the image coordinate and 6 features were $3D$ measurements calculated by stereo triangulation. The 2D measurement noise was assumed to be a bivariate Gaussian process. The uncertainty of an image point measurement can be calculated by fitting a bivariate Gaussian to the local auto-correlation function of the point image [68]. The 3D measurement uncertainty can be easily derived from the image point uncertainty (for further details see [54]). According to the results, the relative transformations between every two positions were calculated. The comparison between the real transformation and the constructed transformation as estimated by the algorithm is given in the following table. As can be seen, the results obtained by our algorithm are close to the real solution with a deviation of up to 0.9 cm in translation and 1.1° in rotation.

	True Solution				Estimated Solution			
	pose A	pose B	pose C	pose D	pose A	pose B	pose C	pose D
pose A		31.0 cm 32.5°	48.6 cm 24.0°	26.3 cm 46.0°		30.2 cm 32.7°	47.7 cm 23.1°	26.8 cm 45.5°
pose B			20.3 cm −8.5°	15.8 cm 13.5°			20.5 cm −9.6°	16.4 cm 12.7°
pose C				35.6 cm 22.0°				35.4 cm 21.1°

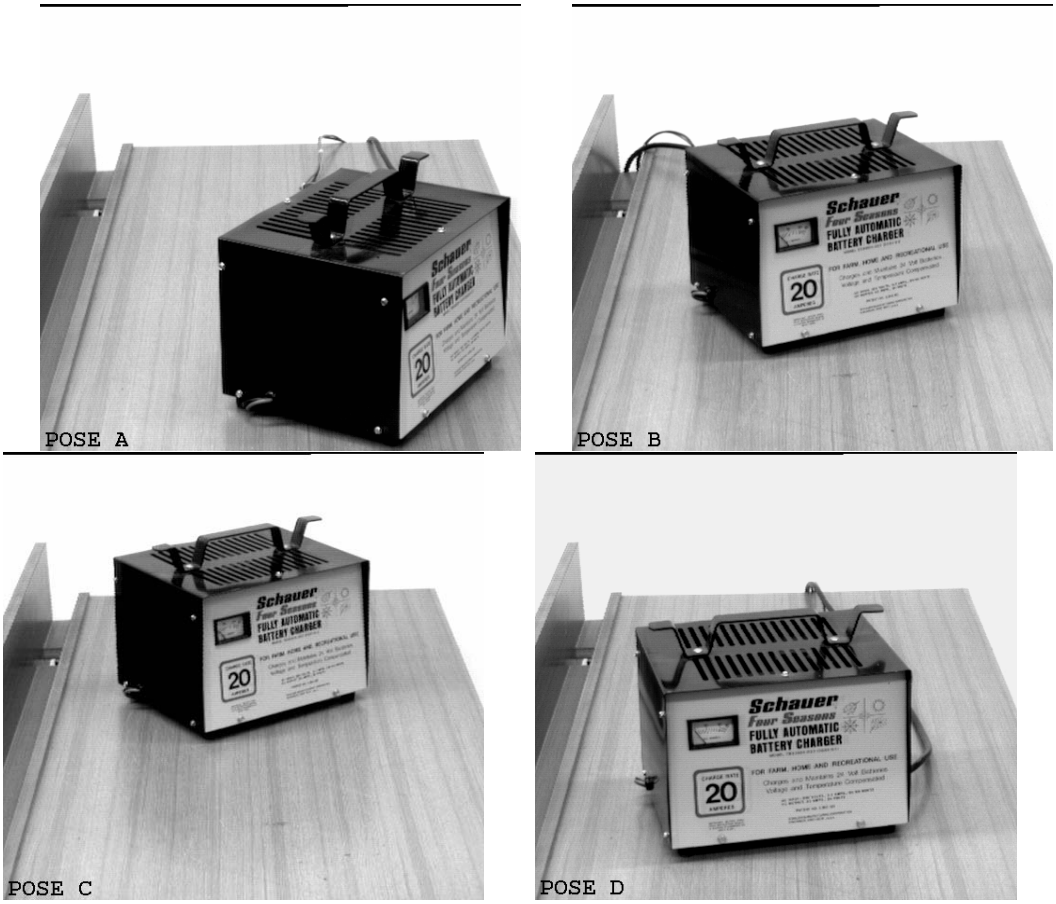

Figure 4.18: Four images of four different positions of an object that were analyzed by our method. Top-left: pose A, top-right: pose B, bottom-left: pose C and bottom-right: pose D.

4.9 Rigid Objects: Conclusion

In this chapter we presented a new approach to estimating the pose of a rigid object in space, where no limitations are imposed on the dimensionality of the measurements and on the type of projection. The main advantages of the suggested approach are as follows:

1. A uniform formulation for all types of measurements allowing simple and efficient fusion of information obtained from different types of sensors.

2. Considering the spatial uncertainty of each measurement, in an explicit manner, enabling optimal exploitation of the available information from the measurements.

3. Fusing the measurements in an incremental process, thus easily incorporated into the matching process which is performed by a pruning-search of the interpretation tree. Additionally, the quality of the matching can be estimated by using statistical tests.

4. The process can easily be parallelized.

5. The process additionally supplies an estimation of the quality of the solution. This quality estimate can assist in determining the number of measurements required for estimating the pose at a given precision.

6. Uncertainties can be easily incorporated into the model. Model uncertainties are an important tool when dealing with object classification into model classes.

Simulations of the described pose estimation process, showed quick and stable convergence of the estimate to the true solution. Simulations of the parallelized process showed similar results. Good and stable solutions were also obtained for real models and images.

The proposed paradigm suggests several extensions to be studied:

1. Many studies in computer vision deal with finding the motion parameters of an object from two or more images taken at different time steps. The method suggested in this chapter can be extended to deal with this motion estimation problem by considering one of the images as the model of the object having infinite uncertainty in the direction of projection. Given a sequence of images, the uncertainty of the model can be incrementally reduced, where every image contributes to improving the model and increasing its precision. This study is currently in progress (by E. Piassetsky and M. Werman at the Hebrew University).

2. The features considered in this chapter were feature points, however a general extension will include additional geometric features such as lines. planes, cylinders etc.

Chapter 5

Pose Estimation of Constrained Models

5.1 Pose Estimation of Articulated and Constrained Models

In the previous chapter, infinite uncertainty was used to fuse measurements of different types in order to determine the pose of a rigid object. In this chapter, we describe the use of zero uncertainty in dealing with a similar problem of evaluating the pose of articulated and constrained objects.

An articulated object is an object composed of a set of rigid components connected at joints which allow certain degrees of freedom. An articulated object having a prismatic joint and a revolute joint can be seen in Figure 5.1. Each model joint enforces a constraint on the position of the body's components, thus, the problem of articulated objects is a special case of the general study of constrained models. We extend the definition of the problem to models that include other constraints such as co-linearity or co-planarity of the model points, angle relationships, etc. The constraints may also include inequality constraints such as limited range of distances between points or limited range of angles. In this section we deal with pose estimation of general constrained models. As in the previous section, we deal here with models consisting of a set of feature points, such as maximum curvature, segment endpoints or corners. The measurements taken on these points are noisy and can be of various types.

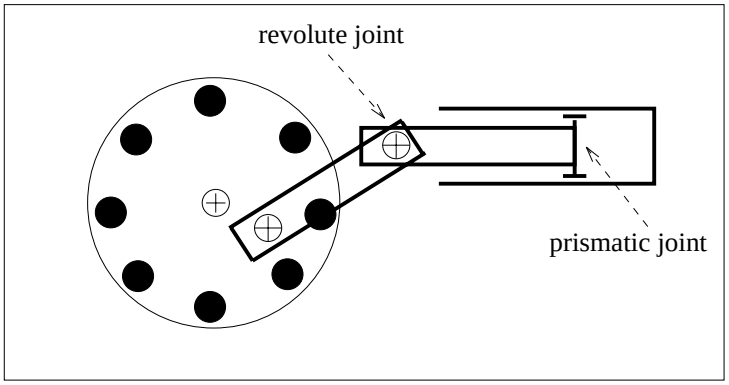

Figure 5.1: An articulated object composed of rigid components connected by two revolute joints and a single prismatic joint.

5.2 Formal Description of the Problem

A *constrained model* M of a $3D$ object consists of a set of rigid components

$$M = \{C_i\}_{i=1\cdots n} \quad .$$

Each component has its own local coordinate system and consists of a set of feature points whose locations are:

$$C_i = \{\mathbf{u}_{i,j}\}_{j=1\cdots m_i} \quad .$$

$\mathbf{u}_{i,j}$ is a 3 dimensional vector, representing the location of the j^{th} point in the i^{th} component and is given in the local coordinate system of C_i. A set of points forming a component is rigid but the collection of components are not rigid.

A *measurement* M' of an articulated object is represented by a collection of noise contaminated measures and their uncertainties:

$$M' = \{(\hat{\mathbf{u}}'_{i,j}, \Lambda_{i,j})\}_{i=1\cdots n \ ; \ j=1\cdots m_i} \quad .$$

$\hat{\mathbf{u}}'_{i,j}$ - is a noise-contaminated measure of the real location-vector $\mathbf{u}'_{i,j}$, associated with the j^{th} measured point of the i^{th} component. $\hat{\mathbf{u}}'_{i,j}$ is represented in a viewer-centered frame of reference. It is possible to have more than one measurement for a model point.
$\Lambda_{i,j}$ - is the covariance matrix depicting the uncertainty in the sensed vector $\hat{\mathbf{u}}'_{i,j}$. We do

not constrain the dimensionality of the measured data but allow it to be $3D$, $2D$ or $1D$, as in the rigid case.

A *matching* between the model M and the measurement M' is a collection of pairs of the form

$$matching = \{\mathbf{u}_{i,j}, (\hat{\mathbf{u}}'_{i,j}, \Lambda_{i,j})\} \quad ,$$

which represents the correspondence between the model points and the measured points. For simplicity we denoted a model point and its matched measurement with the same indices. In the first part of this chapter we assume that the matching is given.

The problem :
Given a model M a measurement M' and the matching between them, estimate for each component C_i a rigid transformation $\mathbf{T}_i$ which optimally maps its feature points onto their measurements, i.e., $\mathbf{T}_i$ is the parameter-vector describing the position of the component C_i in the scene. However, the solution $\{\mathbf{T}_i\}_{i=1\cdots n}$ *must* satisfy a set of constraints

$$\{\psi_k(\mathbf{T}_p, \mathbf{T}_q, \cdots) = 0\} \quad ,$$

which describe the relationships existing between the components. Each constraint can involves a single model component, such as a known location or a known orientation of the component, or several components like a revolute or a prismatic joint between two components, a known distance between components, etc. Each constraint is expressed by an appropriate equation, for example, in an articulated constraint two components, C_p and C_q, are linked at a rotational point whose location is given by $\mathbf{u}'_{p,i}$ in the local coordinates of C_p, and by $\mathbf{u}'_{q,j}$ in the local coordinates of C_q. In such a case the constraint equation will be:

$$T_p(\mathbf{u}_{p,i}) = T_q(\mathbf{u}_{q,j}).$$

where T_i is the transformation function associated with $\mathbf{T}_i$.

As previously mentioned, the model may also consist of inequality constraints of the form $\psi(\mathbf{T}_p, \mathbf{T}_q, \ldots) > 0$. Let us assume, for the moment, that the constraints are restricted to equality constraints, and we will later describe the direct extension of these constraints to inequality constraints.

5.3 Local Constraints Method

In Chapter 2 (Section 2.6) we described two kind of existing methods for estimating the pose of articulated objects; divide and conquer and parametric methods. In these two kind of methods there are no direct consideration of constraints in the estimation process. Either, the constraints are not considered in the divide and conquer methods or they are eliminated, by reducing the number of estimated parameters, in the parametric methods. The method suggested in this paper considers both, measurements and constraints, in the estimation process. The pose of the object parts is estimated to conform optimally with the measurements while satisfying the model constraints. The method we suggest is a general scheme which overcomes the drawbacks of the other methods.

The idea is to treat both measurements and constraints similarly while varying only their associated uncertainty. The constraints are considered as perfect "measurements" with zero uncertainty whereas the measurements themselves (the actual measurements) have uncertainty greater than zero. In other words the actual measurements are considered *soft* constraints whereas the constraints are considered *strong*. The fusion of the actual measurements and the constraints during the pose estimation process is performed using the Kalman filter and it is in accord with [4].

5.3.1 Physical Analogy

In Section 3.4 we gave some physical intuition described the analogy between the K.F. solution and the minimum energy solution of a physical system of springs. We extend this analogy to deal with articulate models. In this case the equivalent physical system is composed of an object model and a collection of springs (see Figure 5.2). These springs attached on one side to the model points and on the other to points in space. Here too, the springs represent the actual measurements where the spring constant denote the uncertainty of the measurement and the anchor points in space represent the measurement values. The solution of the physical system is that which brings the system to minimum energy. This solution is also the solution we wish to achieve and it is the solution we will elaborate in the following using the K.F. equations.

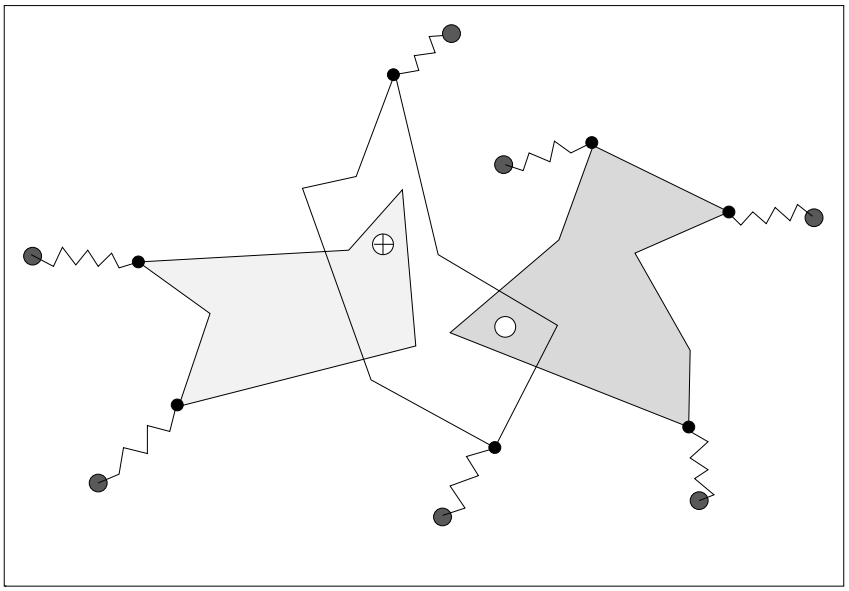

Figure 5.2: An analogous physical system of the estimation problem. Black points represent model points and gray circles represent measurements.

5.4 Solving Constrained Systems Using K.F.

In the following, we will describe three methods for estimating the pose of constrained objects: the dynamic method, the incremental method and the batch method. These methods all use K.F. tools and are theoretically equivalent. Each method has its advantages, making it appropriate for different types of problems. In order to simplify the description of these methods, we first present in detail the solution to these methods for the special case of constrained models consisting of a single point in each component. After laying out the principles and mathematical foundations, we will describe the extension for general constrained models.

5.5 Constrained Objects Having One Point Per Component

The simplest case of a constrained object is where each of its components consists of a single model point. In this case the model is represented by:

$$M = \{C_k\}_{i=1\cdots n}$$

where each component C_k has a single model point whose location is $\mathbf{u}_k$. Without loss of generality, we choose this point to be located at the origin of the local coordinates associated with C_k, i.e: $\mathbf{u}_k = (0,0,0)^t$. Measurements of the locations of the model points are obtained. For simplicity assume n measurements are obtained, $\{(\hat{\mathbf{u}}'_i, \Lambda_i)\}_{i=1\cdots n}$, a single measurement for each model point, represented in the viewer-centered coordinates. Additionally, assume in this case that the measurements are $3D$ data. The latter assumption is due to the inability to induce the $3D$ position of an isolated point from a single $2D$ measurement. The transformation of the k^{th} component, $\mathbf{T}_k$, is composed only of the translation vector $\mathbf{t}_k$ since the rotation part $\tilde{\mathbf{q}}_k$ is irrelevant for an isolated point. Therefore, the general position vector, $\mathbf{T}$, to be estimated in such a case consists of the translation vectors of all the model components:

$$\mathbf{T} = \begin{pmatrix} \mathbf{t}_1 \\ \mathbf{t}_2 \\ \vdots \\ \mathbf{t}_n \end{pmatrix} \quad .$$

Since the model points are located at the origin of the local coordinates the translation vector $\mathbf{t}_k = (x'_k, y'_k, z'_k)^t$ also describes the position of the k^{th} point in the viewer centered frame of reference. However, the evaluated estimation must satisfy a set of constraints:

$$\{\psi_j(\mathbf{T}) = 0\}_{j=1\cdots m} \quad .$$

For the specific case of an articulated object the constraints are:

$$\psi_j(\mathbf{t}_k, \mathbf{t}_l) = \|\mathbf{t}_l - \mathbf{t}_k\|^2 - d^2_{(k,l)} = 0$$

where $d_{(k,l)}$ represents the constant Euclidean distance between two adjacent points, $\mathbf{u}_k$ and $\mathbf{u}_l$, in the object (see Figure 5.3).

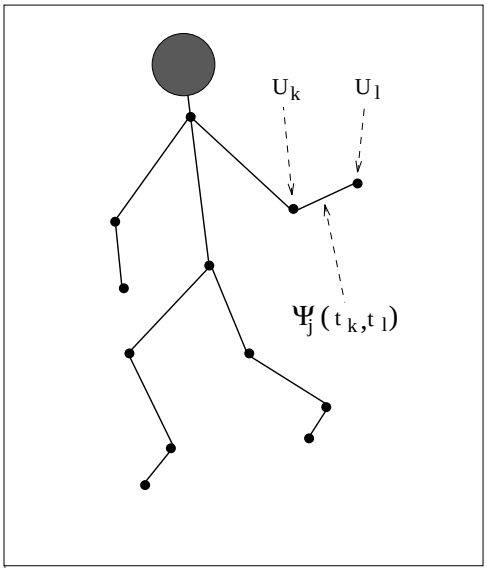

Figure 5.3: An articulated object. The black points represent model features and the segments represent constraints between adjacent points.

5.6 The Batch Approach

As stated, enforcing the model constraints into the pose solution is done by considering the constraints as additional artificial "measurements" having zero uncertainty. The zero uncertainty of these "measurements" assures that the constraints are satisfied in the final solution. The constraint "measurements" will be fused as one big measurement consisting of all the constraints together (batch).

Fusion of the constraints is performed, similar to the fusion of actual measurements, using the E.K.F. tool where the evaluated parameters is the vector $\mathbf{T}$. The fusion will be done by one step of the K.F. fuser followed by local iterations in order to reduce the influence of the linearization. The inputs supplied to the K.F. fuser are the following:

- *A priori* estimate input:
  From the actual measurements we construct an *a priori* estimate of the evaluated transformations:

$$(\hat{\mathbf{T}}^-, \Sigma^-) = \left[\begin{pmatrix} \hat{\mathbf{u}}'_1 \\ \hat{\mathbf{u}}'_2 \\ \vdots \\ \hat{\mathbf{u}}'_n \end{pmatrix}, \begin{pmatrix} \Lambda_1 & & 0 \\ & \ddots & \\ 0 & & \Lambda_n \end{pmatrix} \right] = [\hat{\mathbf{u}}', \Lambda] \tag{5.1}$$

  which takes into consideration *all* the actual measurements.

- Measurement input:
  From the constraint equations we construct a set of artificial perfect "measurements" having zero uncertainty .

- Measurement model input:
  The mathematical relationship between the measurements and the evaluated vector will be a concatenation of all the linear approximations of the constraint equations.

More formally, assume we are given the constraint $\psi_j(\mathbf{T}) = 0$. We regard the measurement $\hat{\mathbf{u}}'$ (Equation 5.1) as an initial approximation of $\mathbf{T}$ and then linearize $\psi_j(\mathbf{T})$ around $\hat{\mathbf{u}}'$ obtaining:

$$\psi_j(\mathbf{T}) = 0 \approx \psi_j(\hat{\mathbf{u}}') + \frac{\partial \psi_j}{\partial \mathbf{T}}(\mathbf{T} - \hat{\mathbf{u}}') \quad .$$

This equation can be rewritten as a linear equation:

$$\mathbf{z}_j = H_j \mathbf{T} \tag{5.2}$$

where

$$\begin{aligned} \mathbf{z}_j &= -\psi_j(\hat{\mathbf{u}}') + (\frac{\partial \psi_j}{\partial \mathbf{T}})\hat{\mathbf{u}}' \\ \text{and} \quad H_j &= \frac{\partial \psi_j}{\partial \mathbf{T}} \end{aligned}$$

The matrix H_j is of dimension $\dim(\psi_j) \times 3n$ representing the linear relationship between $\mathbf{z}_j$ and $\mathbf{T}$. Note that no random noise is added to this equation, so $\mathbf{z}_j$ is a perfect "measurement". The rest of the constraints are similarly linearized and appended to Equation 5.2, so that a vector equation is obtained:

$$\mathbf{z} = H\mathbf{T} \tag{5.3}$$

where

$$\mathbf{z} = \begin{pmatrix} \mathbf{z}_1 \\ \mathbf{z}_2 \\ \vdots \\ \mathbf{z}_m \end{pmatrix} \quad \text{and} \quad H = \begin{bmatrix} H_1 \\ H_2 \\ \vdots \\ H_m \end{bmatrix} \tag{5.4}$$

$\mathbf{z}$ is the "measurement" vector of size $d = \sum \dim(\psi_j)$ with an associated uncertainty matrix of zeroes. H is a $d \times 3n$ matrix that describes the mathematical relationship between $\mathbf{z}$ and the evaluated vector $\mathbf{T}$. In the case where there is more than a single measurement per feature point, these measurements and their uncertainties that were not already considered in the *a priori* estimation will be appended to Equation 5.3 as well.

Given the inputs described above, the estimate for $\mathbf{T}$ obtained from the K.F. equations is (3.2):

$$\hat{\mathbf{T}} = \hat{\mathbf{T}}^- + \Lambda H^t (H \Lambda H^t)^{-1} (\mathbf{z} - H\hat{\mathbf{T}}^-) \quad . \tag{5.5}$$

Multiplying both sides of the equation by H we obtain:

$$H\hat{\mathbf{T}} = H\hat{\mathbf{T}}^- + (H\Lambda H^t)(H\Lambda H^t)^{-1}(\mathbf{z} - H\hat{\mathbf{T}}^-) = \mathbf{z}$$

i.e. the obtained solution indeed satisfies the constraints as defined in Equation 5.3.

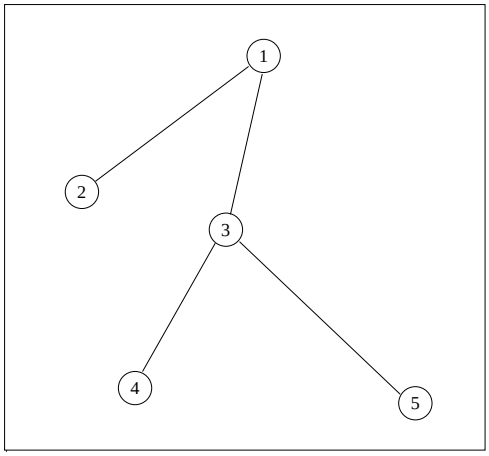

Figure 5.4: An articulated object composed of five single point components and four articulated constraints.

An example of producing the K.F. input for a simple articulated object similar to that in Figure 5.4 is given in Appedix C. Some examples of this articulated object in different positions are shown in Figure 5.5. In these examples the measurements are represented by rectangles having width and length proportional to the s.t.d. of the measurements. The dotted lines connect each measurement to its associated model point.

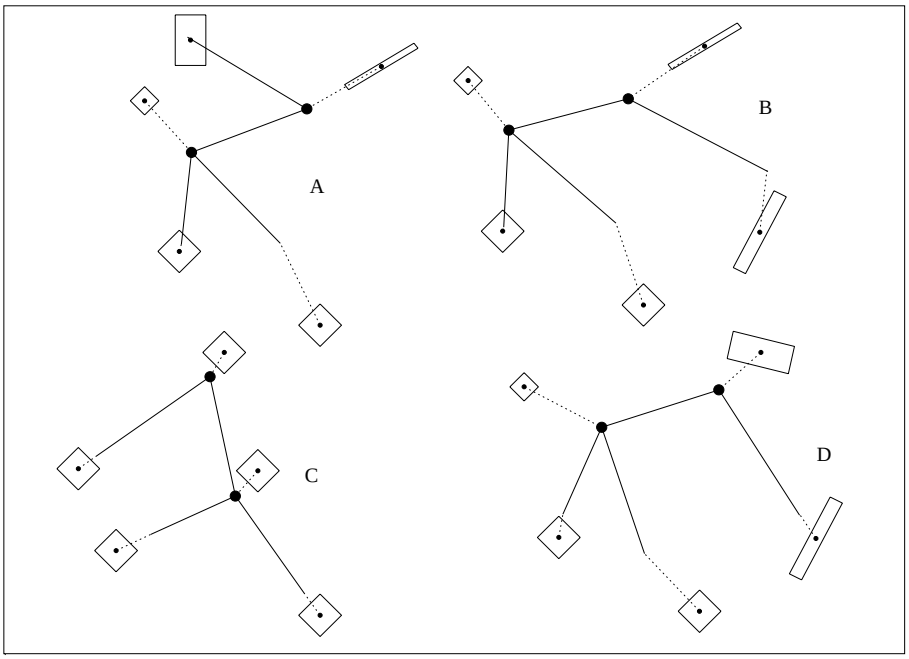

Figure 5.5: Four examples of solutions for simulated articulated object. Joints are represented by black circle and measurements are represented by rectangles. The width and the length of each rectangle is proportional to the s.t.d. of the measurement. The dotted lines connect between a measurement and its associated model point.

Adding An Initial Guess

Using the described inputs, the *a priori* estimate is based on the actual measurements. Although these measurements supply a good *a priori* estimate, in some cases, it seems beneficial to allow the user a possibility of providing an initial guess based on external knowledge. This can be helpful in obtaining a better linearization during the first stages of the process resulting in a more stable computation. Adding an initial guess is easily obtained by slightly varying the input to the K.F. fuser:

- *A priori* estimate input: is taken as the *a priori* estimate supplied by the user and is associated with infinite uncertainty:

$$(\hat{\mathbf{T}}^-, \Sigma^-) = \left[\begin{pmatrix} \hat{\mathbf{t}}_1^0 \\ \hat{\mathbf{t}}_2^0 \\ \vdots \\ \hat{\mathbf{t}}_n^0 \end{pmatrix}, \begin{pmatrix} \infty & & 0 \\ & \ddots & \\ 0 & & \infty \end{pmatrix} \right]$$

- Measurement input: is a concatenation of all the actual measurements and all the constraints "measurements":

$$[\tilde{\mathbf{z}}, \mathrm{cov}\{\tilde{\mathbf{z}}\}] = \left[\begin{pmatrix} \mathbf{z} \\ \hat{\mathbf{u}}' \end{pmatrix}, \begin{pmatrix} 0 & 0 \\ 0 & \Lambda \end{pmatrix} \right]$$

  where $\hat{\mathbf{u}}'$, Λ and $\mathbf{z}$ are the same as defined in Equations 5.1, 5.4 above.

- Measurement model input: is defined in the following equation:

$$\tilde{\mathbf{z}} = \begin{pmatrix} H \\ I \end{pmatrix} \mathbf{T}$$

  where H is defined in Equation 5.3.

This calculation is general enough to include the case where $\mathbf{T}^- = \hat{\mathbf{u}}'$ which is the former case where the actual measurements are taken as the *a priori* estimate.

5.6.1 The Batch Approach: Computational Aspects

The Initial Guess

When dealing with the batch K.F. solution to the problem of constrained systems, one should note that similar to all other methods of constrained optimization [34] the batch K.F. method may erroneously converge to a local minima. Thus, here too the solution depends on the initial guess, i.e. on the *a priori* estimate supplied to the filter.

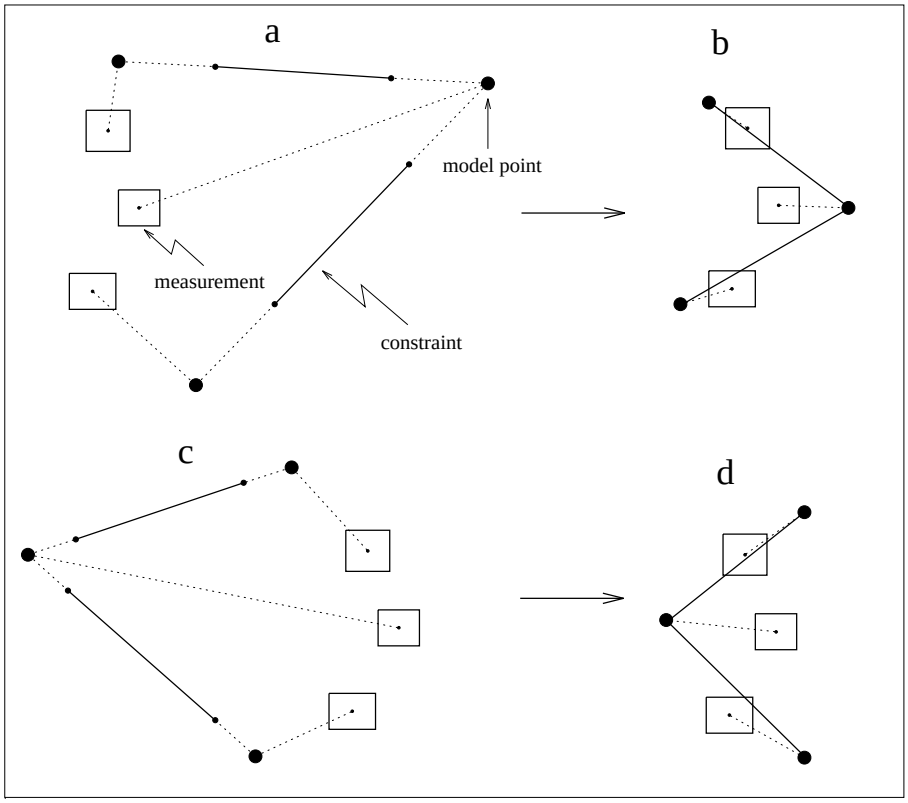

Figure 5.6: Two examples of *a priori* estimates (a and c) and their corresponding solutions (b and d). Model points are represented by black circles, measurements and their uncertainty by rectangles and distance constraints by bold lines. The dotted lines connect between a measurement and its associated model point.

We may obtain some intuition on the character of these local minima from the analogous physical system of springs described in Section 3.4. Initializing the springs of the

physical system to a state close to a local minima may cause the springs system to converge and stabilize at that local minima. The energy of unconstrained system of springs (a physical system consists only of springs) does not have local minima since the energy depends quadratically on the deformations of the springs. Therefore local minima are due to the constraints added to the system and the chances of adding local minima to the system, increases with the number of constraints. However, we emphasize that from our experience of simulations of constrained systems, the system does converge to the global minima when a reasonable *a priori* estimate is given, and we find that its basin of attraction is quite large.

Figure 5.6 display results of simulations of a constrained system, for two *a priori* estimates. It can be shown that the *a priori* estimate input influences the final solution of the system.

Stability

Several problems may prevent the local iterations from converging or cause the solution to oscillate about the true solution. These problems arise from two main sources which were mentioned previously:

1. The fusion of perfect measurements with noisy measurements creates an ill-posed matrix which must be inverted during the filtering process. Inversion of such matrices creates computational imprecision during the process.

2. The linear approximation of the non linear measurement model (which includes the constraints) may create imprecision which can prevent convergence. This is specifically true when the *a priori* estimate is distant from the true solution (since linearization is done around this *a priori* estimate).

Regarding the fusion of perfect and noisy measurements, several techniques are known to deal with this problem [55]. These techniques ensure the separation of perfect and noisy measurements by transforming the parameter space. From our experience of simulated constrained systems, we find that quite often the process oscillates about the true solution, specifically when the actual measurements are distant from the true measurements.

An example can be seen in Figure 5.7. This phenomenon is similar to the oscillations of a physical system of springs about a stable state in a frictionless environment. We extend the batch K.F. process to two types of methods which deal efficiently with this problem:

Method I: Adding a damping force

Incorporating a damping force is a common technique in gradient-descent methods. There is a known trade-off between the rate of convergence and the reliability of the convergence, as a function of the damping strength added to the system. Adding a damping force to the K.F. process is simple in the case where the *a priori* estimate input is a user defined initial guess with an associated infinite uncertainty (Section 5.6). In this case, the *a priori* estimate $\hat{\mathbf{T}}^-$ is given a finite uncertainty rather than an infinite one. Additionally, during the iterative process, we continuously update the *a priori* estimate input by taking the resulting estimate obtained at the previous step.

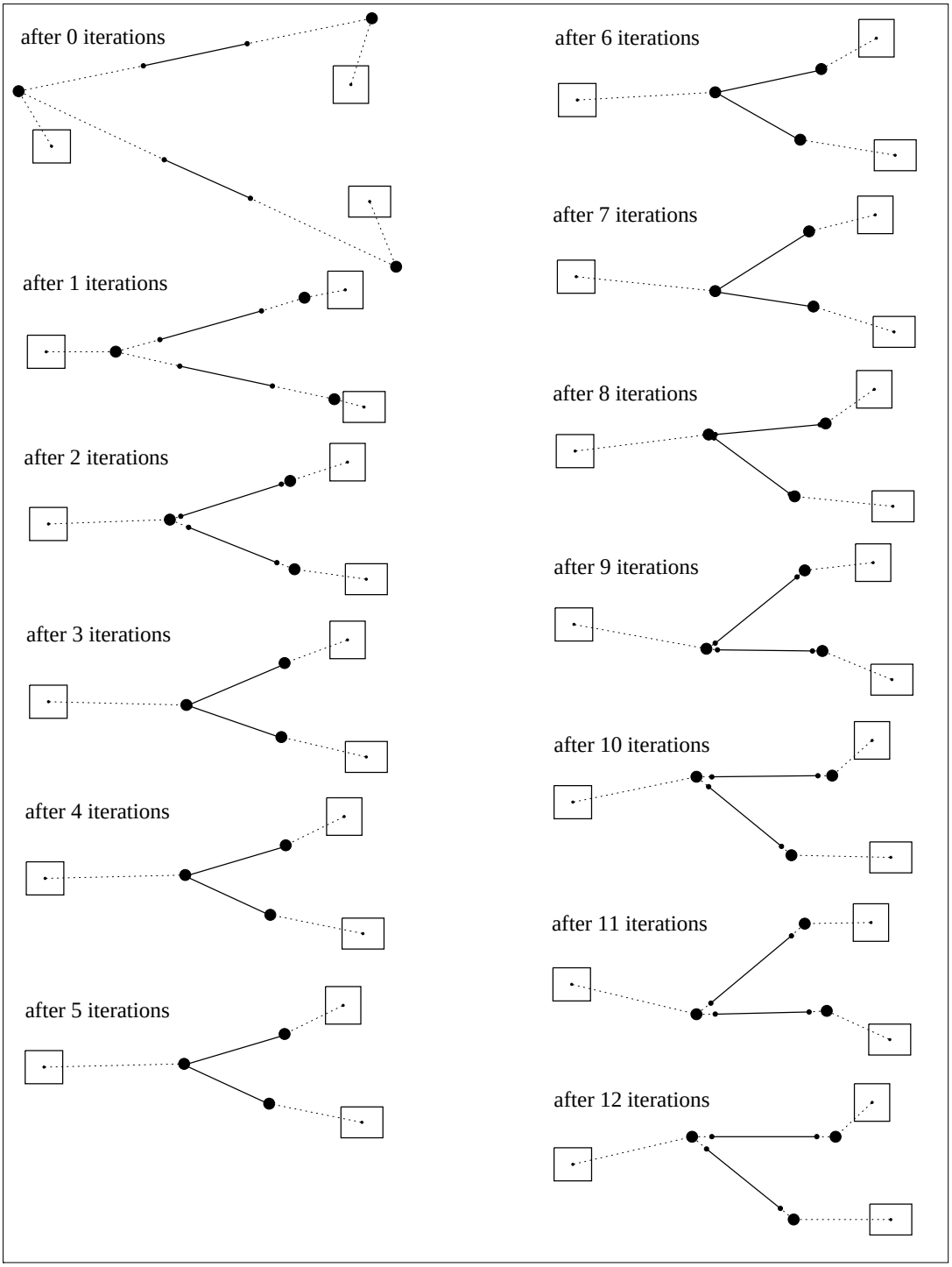

Figure 5.7: An example of local oscillations around the true solution. During the first few iterations the process converges toward the real solution however in further iterations the process oscillates around the solution but does not converge to it.

This method adds a damping force, in the direction of the *a priori* estimate. This is similar to convergence of a physical system containing friction. As the uncertainty of the *a priori* estimate is smaller, the damping force is larger. We should note that Lowe [52, 51], in dealing with pose estimation of articulated objects using minimization of free parameters, also includes a damping factor to stabilize the solution. Additionally he discusses the relation between the strength of the damping force, and the rate and assurance of the convergence. This relation was defined by Levenberg and Marquardt [46, 53] and holds in our case as well: When the damping factor is small (i.e. the *a priori* uncertainty is large), the process is similar to the Newton iteration, which ensures fast convergence but a small basin of attraction. As the damping factor increases (smaller *a priori* uncertainty), the process is more similar to regular gradient-descent methods resulting in decreasing incremental steps but increasing the basin of attraction. Marquardt [53] suggest a simple algorithm for adjusting the damping strength at each step. This algorithm can be easily implemented in our method.

Method II: Flexibility of the constraints

In principle, given no constraints the process converges rapidly to the measurement positions. We can bring a constrained system to a similar situation by assigning infinite uncertainty to the constraint "measurements". Following this idea, in this method, we initially set high uncertainty values to the constraint "measurements" and progressively decrease the value at each iterative stage. The process continues until uncertainty zero is reached which is the value that should be associated with the constraint "measurements". The physical analogy of this paradigm is to slowly change the stiffness of the physical joints in the system so that initially the joints are flexible and unrestrained. As the process continues the joints become less flexible until finally become totally rigid (zero uncertainty). The system in such a process is in a stable state at each iterative stage and does not produce oscillations.

Implementing these two methods gave very good results and allowed a smooth and stable convergence of the process. No significant difference was found between the convergence of these two approaches. Figures 5.8 and 5.9 show examples of the convergence of a simulated constrained system, according to the above two methods. Figure 5.8 shows

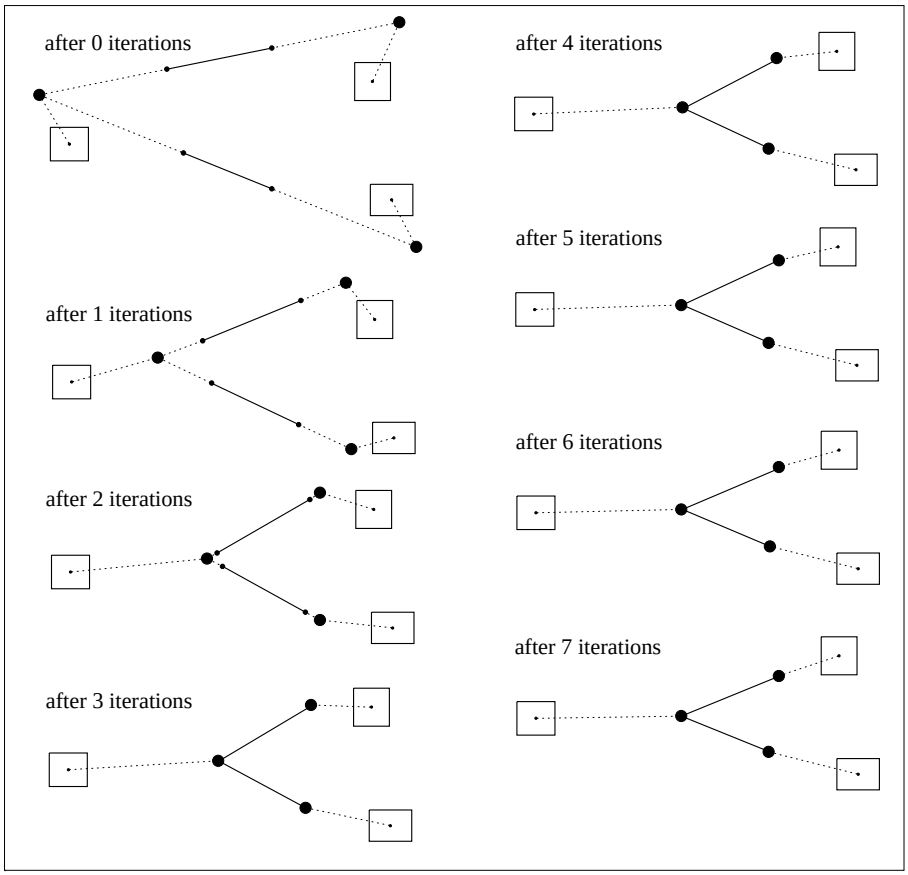

Figure 5.8: Adding a damping force (method I): process convergence to the solution and no oscillations occur.

convergence with adding a damping force and Figure 5.9 shows convergence with flexible constraints.

Existence of a Solution

Up till now we assumed that a solution exists to the constrained system. However, this assumption is not always true; there are cases where the constraints conflict with one another and no solution exists. Furthermore, even when no conflicts arise in the system, some cases can not be solved in the usual method. In order to analyze those cases in which the system does not have a solution, let us consider the implementation (described in Section 5.6) where the actual measurements are supplied to the filter as an *a priori*

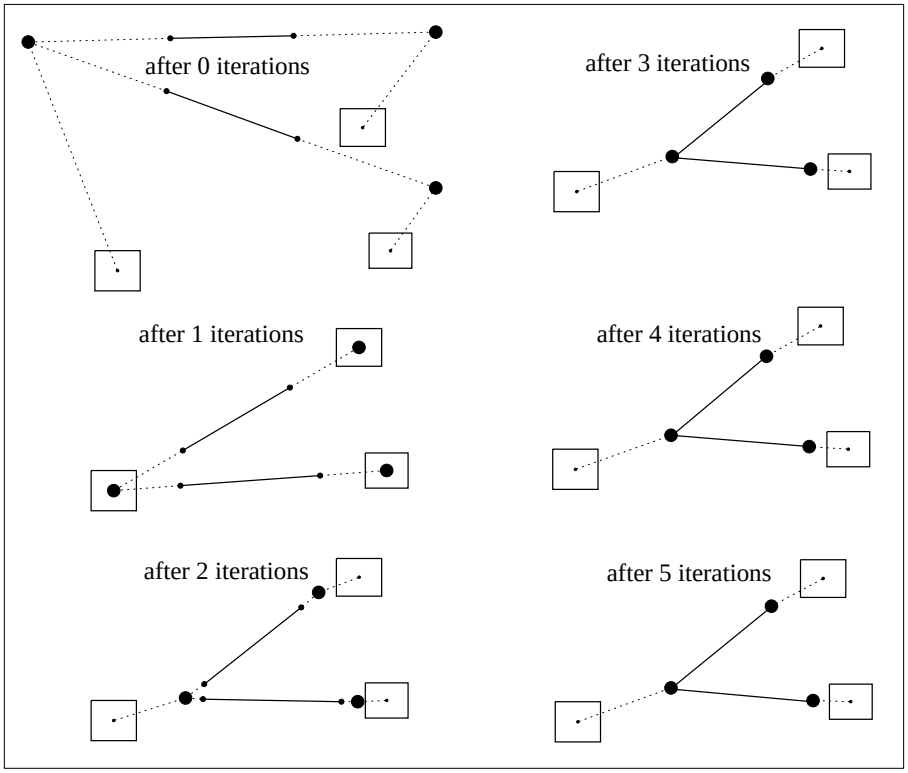

Figure 5.9: Changing the constraints to be flexible constraints (method II): process convergence to the solution and no oscillations occur.

estimate input and the constraints are given as "measurements" with zero uncertainty. In this implementation, the K.F. equations involve inversion of the matrix: $H\Lambda H^t$ (see Eqs. 3.2). The process will fail when this matrix is singular. Thus, it is sufficient to study the singularity cases of this matrix. H is a $d\times 3n$ matrix, Λ is an $3n\times 3n$ matrix and $H\Lambda H^t$ is a $d \times d$ matrix. Suppose first that $m = 3n < d$ in which case $Rank(H) \leq m$, therefore also $Rank(H\Lambda H^t) \leq m < d$. But $H\Lambda H^t$ is a $d \times d$ matrix, therefore it is singular. In other words, there is no solution when the system is over-constrained (more constraint equations than unknowns). On the other hand, suppose that $d < m$ but some of the constraints are depend on other constraints. This means that if constraint linearization is performed at the true solution, some rows of H will be linearly dependent on other rows. In this case $H\Lambda H^t$ is singular since $Rank(H) < d$ hence $Rank(H\Lambda H^t) < d$. The cases when $H\Lambda H^t$ can not be inverted can easily be identified in the course of the

solution process. When this happens, we can try an additional linearization point, since singularity may be the result of a special configuration of the linearization point. When this fails and there is a reason to suspect that the constraints are not contradicting we can attempt to find a maximal set of rows in H which are linearly independent and try to solve the resulting system.

Complexity

The complexity of each iteration of the K.F. depends on the dimension d of the measurement vector and on the dimension $m = 3n$ of the state vector (Section 3.5). In our case m is proportional to the number of model points and d depends on the type of implementation: If the measurement input consists only of constraint "measurements" we have $d = \sum_i \dim(\psi_i)$. In the implementations which add a user defined *a priori* estimate to the system the measurement input also include the actual measurements, and then $d = \sum_i \dim(\psi_i) + \dim(T)$. However, the system must not be over-constrained, so $\sum_i \dim(\psi_i) \leq \dim(T)$. Thus, $d \leq 2\dim(T)$ and the upper bound of the complexity of each iteration is $O(n^3)$.

The rate of convergence (number of iterations) also depends on the implementations: If no damping factor is added to the system, the convergence rate of the system is equivalent to that of the Newton iterations which is quadratic near the solution [71]. When a damping factor is use to stabilize the process, the rate of convergence decreases as the strength of the damping increases.

Run time can be reduced if the matrix $M = H\Lambda H^t$ is not inverted at each iteration. Instead, one can use an approximation of M obtained from previous stages. This approach is appropriate in those implementations where the rate of convergence is low (due to high damping factor) so the state-vector T, and accordingly the matrix M, do not change greatly between iterations (T is the linearization point producing M).

5.6.2 The Batch Approach: Discussion

We described the batch approach for solving systems which include probabilistic measurements and constraints. This method is attractive because of its simple implementation and because of the wide variety of problems which it can solve, including systems with cyclic constraints and constraints depending on several model points simultaneously.

The batch method has two main disadvantages:

1. The method must assume that the matching between model points and measurements is given. Thus, this method is appropriate mainly for pose estimation as a verification process which follows an interpretation process [69, 42, 28].

2. Complexity of the method is high. The method does not exploit the fact that most of the constraints are associated with only a few model points and that, in practice, the constraint matrix H is very sparse.

5.7 The Incremental Approach

In the incremental method, as in the batch method, the state-vector T is a composition of all the pose parameters of the model points, however in this method the actual measurements and the constraint "measurements" are fused sequentially. At each step k, the current estimate $(\hat{\mathbf{T}}^k, \Sigma^k)$ is updated to be $(\hat{\mathbf{T}}^{k+1}, \Sigma^{k+1})$ by fusing a single measurement $(\hat{\mathbf{u}}'_k, \Lambda_k)$ or a constraint $(\mathbf{z}_k, 0)$. The sequential fusion of the measurements is possible due to the assumption that there is no correlation between the noise of different measurements (i.e. Λ in Eq. 5.1 is diagonal). The advantage of the incremental method over the batch method is the ability of the former to easily incorporate a matching (interpretation) process into the estimation process, however, their complexity is the same.

5.7.1 The Measurement Interpretation

Till now, we assumed the matching was given, however this assumption is not reasonable in many applications specifically in the context of recognition. The incremental process can solve this problem since the interpretation process and the pose estimation process can be performed simultaneously in a manner similar to that described for rigid objects. Thus a partial interpretation can assist in eliminating irrelevant matches.

Suppose we want to match the measurement $(\hat{\mathbf{u}}'_i, \Lambda_i)$ with the j^{th} model point. From the current estimate $(\mathbf{T}^{cur}, \Sigma^{cur})$, we extract an estimate of the location $\mathbf{u}_j$: $(\hat{\mathbf{t}}_j^{cur}, \Sigma_j^{cur})$ and evaluate the Mahanalobis distance between $\hat{\mathbf{t}}_j^{cur}$ and $\hat{\mathbf{u}}'_i$:

$$\delta = (\hat{\mathbf{t}}_j^{cur} - \hat{\mathbf{u}}'_i)(\Sigma_j^{cur} + \Lambda_i)^{-1}(\hat{\mathbf{t}}_j^{cur} - \hat{\mathbf{u}}'_i)^t$$

If δ is greater than a predefined threshold, the match is rejected. The greater the number of measurements and constraints fused prior to the match, the more precise is the estimate $\hat{\mathbf{t}}_j^{cur}$ and the elimination of irrelevant measurements is more effective. Therefore, there is great importance, in this method, to the order of the points being fused (matched) since before matching the j^{th} model point, we would like the system to obtain as much information as possible on the location estimate $\hat{\mathbf{t}}_j$ so that the match verification is significant. Additionally, before fusing a non-linear constraint we would like to obtain as much information as possible on the pose of the points associated with this constraint.

The greater the precision of the pose estimate of these points, the greater the quality of the linearization of the non-linear constraint equation. Thus, at each step of the process the next point to be matched should be one associated with previously matched points through constraints, so that previous information can be exploited. The following algorithm follows this idea.

Denote by ψ_k $(k = 1 \ldots m)$ the constraints of the model and by $point(\psi_k)$ the set of points on which ψ_k depends. The order of fusion of the measurements and the constraints is obtained from the following algorithm:

1. $FusedPoints = \emptyset$
2. $ConstList = \{\psi_1, \ldots \psi_m\}$
3. **while** $ConstList \neq \emptyset$ **do**
   (a) **for each** $(\psi_k \in ConstList$ s.t. $point(\psi_k) \in FusedPoints)$ **do**
      i. fuse ψ_k and delete ψ_k from $ConstList$
   (b) **if** there exists $\psi_k \in ConstList$ s.t. $(point(\psi_k) \cap FusedPoints) \neq \emptyset$ **do**
      i. fuse ψ_k and delete ψ_k from $ConstList$
      ii. find a set of measurements $\{\hat{\mathbf{u}}'_k\}$ that match the set of model points $(point(\psi_k) - FusedPoints)$
      iii. fuse the set of measurements $\{\hat{\mathbf{u}}'_k\}$ and add $(point(\psi_k) - FusedPoints)$ to $FusedPoints$
   (c) **else** select an arbitrary $\psi_k \in ConstList$ and **do** steps i-iii in (b).

In the case where the constraint ψ_k is non-linear, the set of matched measurements $\hat{\mathbf{u}}'_k$ are used as linearization points during the fusion of ψ_k. Thus if the match was rejected, ψ_k must be relinearized about the new matched measurements and ψ_k must be fused again.

In the case where a good match for the model point $\mathbf{u}_j$ can not be found due to occlusion or inability to obtain information about certain interest points in the image. In these cases we synthesize an artificial measurement for the model point and associate

it with an infinite uncertainty so that its influence on the rest of the process will be minimal. This scheme can also be helpful when we want to fuse a constraint ψ_k where some of its associated points $point(\psi_k)$ are unavailable.

5.7.2 The Incremental Approach: Computational Aspects

The computational process which follows the incremental method, uses the K.F. fuser at each step. The local iterations used in the batch method to improve the linear approximations, is applied in the incremental method as well. Additionally, the incremental method applies *global iterations* (see [43]) which continuously repeat the full estimation process where the linearization of the non-linear equations are performed initially about the solution obtained at the end of the previous global iteration. The global iterations in the incremental method have the same effect as the local iterations in the batch method, however the former will require fewer iterations since the additional local iterations assist in the convergence.

Since every fusion of a single measurement requires time complexity of $O(n^2)$ (the dimensionality d of each measurement is constant in this case), the complexity of a single global iteration is, similar to the batch method, $O(n^3)$ (assuming the number of local iterations is restricted).

5.8 The Dynamic Approach

The dynamic method, described in this chapter is advantageous in that it has a lower time complexity and still allows interpretation of the measurement during the estimation process. However, the implementation of this method is more complicated than the previously described methods.

In this method the evaluated vector $\mathbf{T}$ is decomposed into its components $\{\mathbf{t}_k\}$ where each position vector $\mathbf{t}_k$ is evaluated separately. The idea is to consider the position vector $\mathbf{t}_k$ as a *dynamic* vector which changes during time and follows a Markovian like process:

$$\mathbf{t}_k \rightarrow \mathbf{t}_{k+1} \rightarrow \mathbf{t}_{k+2} \rightarrow \cdots$$

The information on position $\mathbf{t}_{k+1}$ obtained from position $\mathbf{t}_k$ is derived from the constraint that exists between them.

Initially we describe in detail the "dynamic" solution for a chain-like model and later on we expand the solution to deal with more general models.

5.8.1 Dynamic Solution for a Chain Formation Model

A chain model is a model in which every position vector $\mathbf{t}_k$ appears in exactly two constraints (with the exception of the two end points):

$$\psi_{k-1}(\mathbf{t}_{k-1}, \mathbf{t}_k) = 0$$
$$\psi_k(\mathbf{t}_k, \mathbf{t}_{k+1}) = 0$$

and each constraint ψ_k depends on only two model points $\mathbf{t}_k$ and $\mathbf{t}_{k-1}$ (see Figure 5.10).

In this case we use the dynamic K.F. process for estimating a dynamic parameter vector; at each step k, we estimate $\mathbf{t}_k$ according to:

- The information obtained from the previous position vector $\mathbf{t}_{k-1}$.
- The constraint ψ_{k-1} between $\mathbf{t}_{k-1}$ and $\mathbf{t}_k$.
- The measurements $\hat{\mathbf{u}}'_k$ of the k^{th} point.

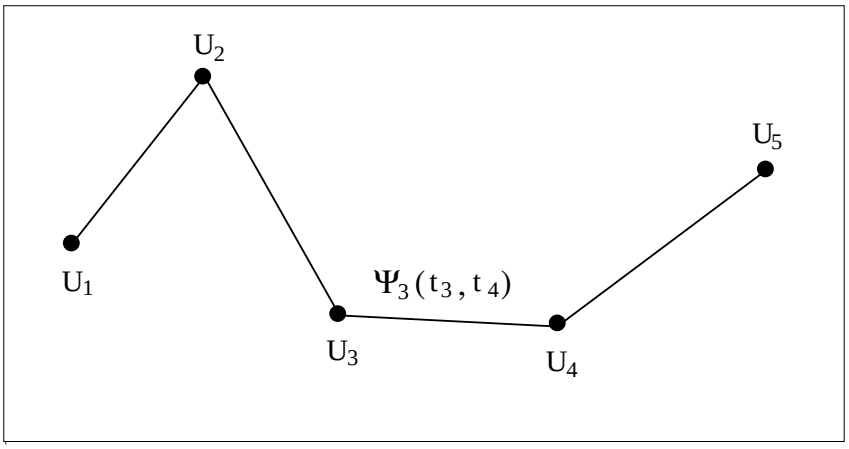

Figure 5.10: Chain like model. Each point has no more than two constraints.

The K.F. process for estimating a dynamic parameter vector can be described as a two-stage-process (see Figure 5.11). The first stage is a *predictor* which supplies, at each step k, an *a priori* estimate of $\mathbf{t}_k$, based on the $k-1$ previous measurements and the constraints existing between them. This *a priori* estimate will be denoted $\hat{\mathbf{t}}_{k|k-1}$. The predictor additionally supplies the uncertainty associated with $\hat{\mathbf{t}}_{k|k-1}$ denoted $\Sigma_{k|k-1}$. The second stage is the K.F. *fuser* which is similar to that described in Section 3.1. This stage fuses the measurement $\hat{\mathbf{u}}'_k$ with the *a priori* estimate $\mathbf{t}_{k|k-1}$ in order to obtain an updated estimate $\mathbf{t}_{k|k}$ and uncertainty $\Sigma_{k|k}$.

The Predictor: Fusing the Constraints

The general K.F. predictor assumes that the dynamics of the evaluated parameters are given explicitly (e.g. $\mathbf{t}_k = F\mathbf{t}_{k-1}$). However, in our case the *a priori* estimate of $\mathbf{t}_k$ is evaluated using the constraint $\psi_{k-1}(\mathbf{t}_{k-1}, \mathbf{t}_k)$. Thus, the information that the constraint supplies must be fused at the predictor stage. The prediction process will also be performed using the K.F. fuser as follows:
We regard the measurement $\hat{\mathbf{u}}'_k$ as an approximation of $\mathbf{t}_k$ and then linearize ψ_{k-1} around $(\hat{\mathbf{t}}_{k-1|k-1}, \hat{\mathbf{u}}'_k)$ obtaining:

$$-\psi_{k-1}(\hat{\mathbf{t}}_{k-1|k-1}, \hat{\mathbf{u}}'_k) + \left(\frac{\partial \psi_{k-1}}{\partial \mathbf{t}_{k-1}}\right) \hat{\mathbf{t}}_{k-1|k-1} + \left(\frac{\partial \psi_{k-1}}{\partial \mathbf{t}_k}\right) \hat{\mathbf{u}}'_k = \left[\frac{\partial \psi_{k-1}}{\partial \mathbf{t}_{k-1}}, \frac{\partial \psi_{k-1}}{\partial \mathbf{t}_k}\right] \begin{pmatrix} \mathbf{t}_{k-1} \\ \mathbf{t}_k \end{pmatrix}$$

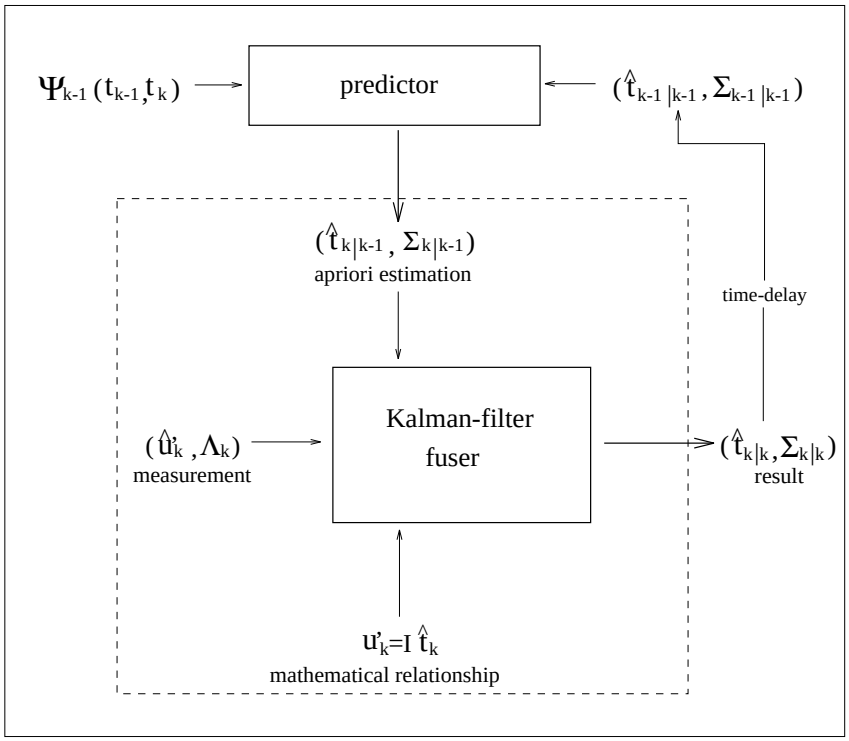

Figure 5.11: Kalman filter for evaluating dynamic parameters. The predictor and the fuser.

This equation can be rewritten as a linear equation:

$$m_k = H \begin{pmatrix} \mathbf{t}_{k-1} \\ \mathbf{t}_k \end{pmatrix} \tag{5.6}$$

where

$$\begin{aligned} m_k &= -\psi_{k-1}(\hat{\mathbf{t}}_{k-1|k-1}, \hat{\mathbf{u}}'_k) + \left(\frac{\partial \psi_{k-1}}{\partial \mathbf{t}_{k-1}}\right) \hat{\mathbf{t}}_{k-1|k-1} + \left(\frac{\partial \psi_{k-1}}{\partial \mathbf{t}_k}\right) \hat{\mathbf{u}}'_k \\ \text{and} \quad H &= \left[\frac{\partial \psi_{k-1}}{\partial \mathbf{t}_{k-1}}, \frac{\partial \psi_{k-1}}{\partial \mathbf{t}_k}\right] \quad . \end{aligned}$$

m_k is regarded as the measurement input with zero uncertainty and Equation 5.6 is the measurement model input. The predictor assumes $\begin{pmatrix} \mathbf{t}_{k-1} \\ \mathbf{t}_k \end{pmatrix}$ as the state vector to be estimated. The *a priori* estimate input given to the predictor is:

$$\left[\begin{pmatrix} \hat{\mathbf{t}}_{k-1|k-1} \\ \hat{\mathbf{u}}'_k \end{pmatrix}, \begin{pmatrix} \Sigma_{k-1|k-1} & 0 \\ 0 & \infty \end{pmatrix} \right] \quad ,$$

where $\hat{\mathbf{t}}_{k-1|k-1}$ and $\Sigma_{k-1|k-1}$ are the estimate and uncertainty obtained at the previous step for $\mathbf{t}_{k-1}$. The uncertainty ∞ associated with the *a priori* estimate of $\mathbf{t}_k$ will eliminate the influence of this input (the measurement $\hat{\mathbf{u}}'_k$) in the final solution since it will be considered again in the fuser stage.

Inserting the *a priori* estimate and the "measurement" given in Equation 5.6 into the K.F. updating equations, gives an updated estimate for $\begin{pmatrix} \mathbf{t}_{k-1} \\ \mathbf{t}_k \end{pmatrix}$:

$$\left[\begin{pmatrix} \hat{\mathbf{t}}^+_{k-1|k-1} \\ \hat{\mathbf{t}}^+_{k|k-1} \end{pmatrix}, \begin{pmatrix} \Sigma^+_{k-1|k-1} & \cdots \\ \cdots & \Sigma^+_{k|k-1} \end{pmatrix} \right] \quad . \tag{5.7}$$

The estimation $\hat{\mathbf{t}}^+_{k|k-1}$ and its uncertainty $\Sigma^+_{k|k-1}$ which appear in Equation 5.7 are the output of the predictor and serve as an *a priori* estimate of $\mathbf{t}_k$. This prediction is produced using only the constraint ψ_{k-1} and the information obtained from $\hat{\mathbf{t}}_{k-1|k-1}$. The next stage is to fuse the current measurement with the *a priori* estimate supplied by the predictor. This will be performed by the *fuser*.

The Fuser: Fusing the Measurements

Given an *a priori* estimate of $\mathbf{t}_k$ from the predictor, its fusion with the k^{th} measurement $(\hat{\mathbf{u}}'_k, \Lambda_k)$ is again performed using the K.F. fuser. Since the measurement input is a measurement of the evaluated vector $\mathbf{t}_k$, the measurement model is simply:

$$\mathbf{u}'_k = I\mathbf{t}_k$$

where I is the identity matrix. In the case where several measurements of $\mathbf{u}'_k$ are given, we consider them as a single "measurement" which is a concatenation of all the measurements. Similarly, their relating equations are concatenated into a single vector equation.

The Smoother: Fusing the Future Measurements

We described how to obtain the position $\mathbf{t}_k$ of the k^{th} model point from the preceding measurements $\{\hat{\mathbf{u}}'_i\}_{i=1\cdots k}$ where the measurement information propagated through the constraints. It is obvious that the estimate $\hat{\mathbf{t}}_{k|k}$ is not the final estimate since subsequent

measurements $\{\hat{\mathbf{u}}'_i\}_{i=k+1\cdots n}$ also influence the estimate of $\mathbf{t}_k$. In order to take into consideration all the measurements we use the *optimal smoothing* (O.S) (see [55]) which is an extension of the K.F. The O.S. consists of two passes (see Figure 5.12):
The first pass (forward-pass) estimates each position vector $\mathbf{t}_k$, starting from $\mathbf{t}_1$ and end-

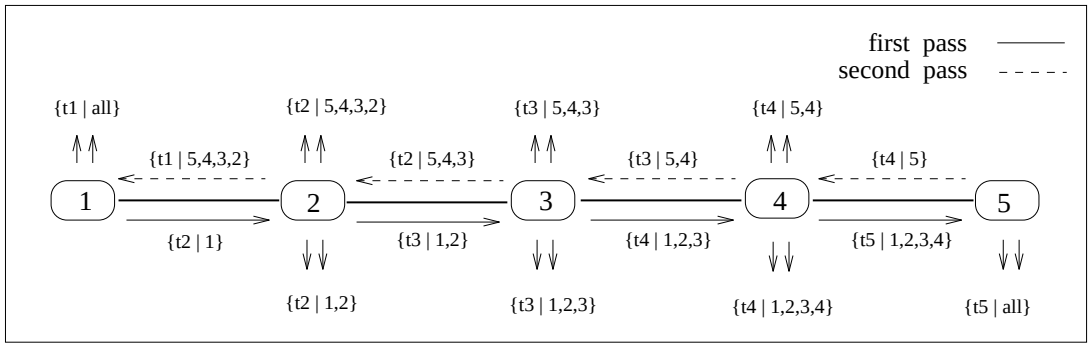

Figure 5.12: Two passes of the optimal smoother. The pose of an articulated model of 5 point-components is evaluated by a forward pass (bold arrows) and a backward pass (dashed arrows).

ing with $\mathbf{t}_n$, based on the previous measurements as described above. At this stage all the *a priori* estimates supplied by the predictor are stored, i.e. for each $\mathbf{t}_k$ the prediction $\hat{\mathbf{t}}^{forward}_{k|k-1}$ is stored.
The second pass (backward-pass) performs the same estimation as the first pass, only reversing the order of the evaluated vector, starting from $\mathbf{t}_n$ and ending with $\mathbf{t}_1$. That is, for each $\mathbf{t}_k$ we obtain an *a priori* estimate $\hat{\mathbf{t}}^{backward}_{k|k+1}$ based on the measurements following $\mathbf{t}_k$, i.e. the measurements $\{\hat{\mathbf{u}}'_i\}_{i=k+1\cdots n}$. The final estimate of $\mathbf{t}_k$ is obtained by fusing the two *a priori* estimates (from the two passes) with the current measurement $\hat{\mathbf{u}}'_k$. This can also be done using the K.F. fuser. Note, that the result of fusing the current measurement with anyone of the two *a priori* estimates already exists (from the forward/backward passes) so only one fusion is required, for example: fusion of $\hat{\mathbf{t}}^{forward}_{k|k}$ with the *a priori* estimate $\hat{\mathbf{t}}^{backward}_{k|k+1}$

In order to formulate a general algorithm to estimate the pose of an articulated object we represent the data by a directional graph $G = (V, E)$ whose nodes V represent the model point and a constraint ψ_{k-1} is represented by two directional edges $(k-1, k) \in V$ and $(k, k-1) \in V$.

The algorithm:

1. perform two passes:
   (a) *forward pass* - pass sequentially over the graph nodes starting from $\mathbf{t}_1$, ending with $\mathbf{t}_n$ and setting for each directional edge $(\mathbf{t}_{k-1}, \mathbf{t}_k)$ the estimate $\hat{\mathbf{t}}_{k|k-1}^{forward}$.
   (b) *backward pass* - pass sequentially over the graph nodes starting from $\mathbf{t}_n$, ending with $\mathbf{t}_1$ and setting for each edge $(\mathbf{t}_{k+1}, \mathbf{t}_k)$ the estimate $\mathbf{t}_{k|k+1}^{backward}$.
2. At the end of the two passes: At each node, fuse all the estimates entering that node from the directional edges with the measurement corresponding to that node.

Formulating the algorithm as given above will help us to formulate the solution for general structures of constrained objects, as will be elaborated in Sections 5.8.2 and 5.8.3.

As in the incremental method, here too, local and global iterations must be used in order to improve the linear approximations of the constraint equations. The complexity of each call to the K.F. fuser during the dynamic method is constant because both the measurement vector and the state vector have bounded dimensions. Thus the complexity of the whole algorithm for the chain case is $O(n)$.

An example for applying the two passes for a chain object such as depicted in Figure 5.12 is given in Figure 5.13.

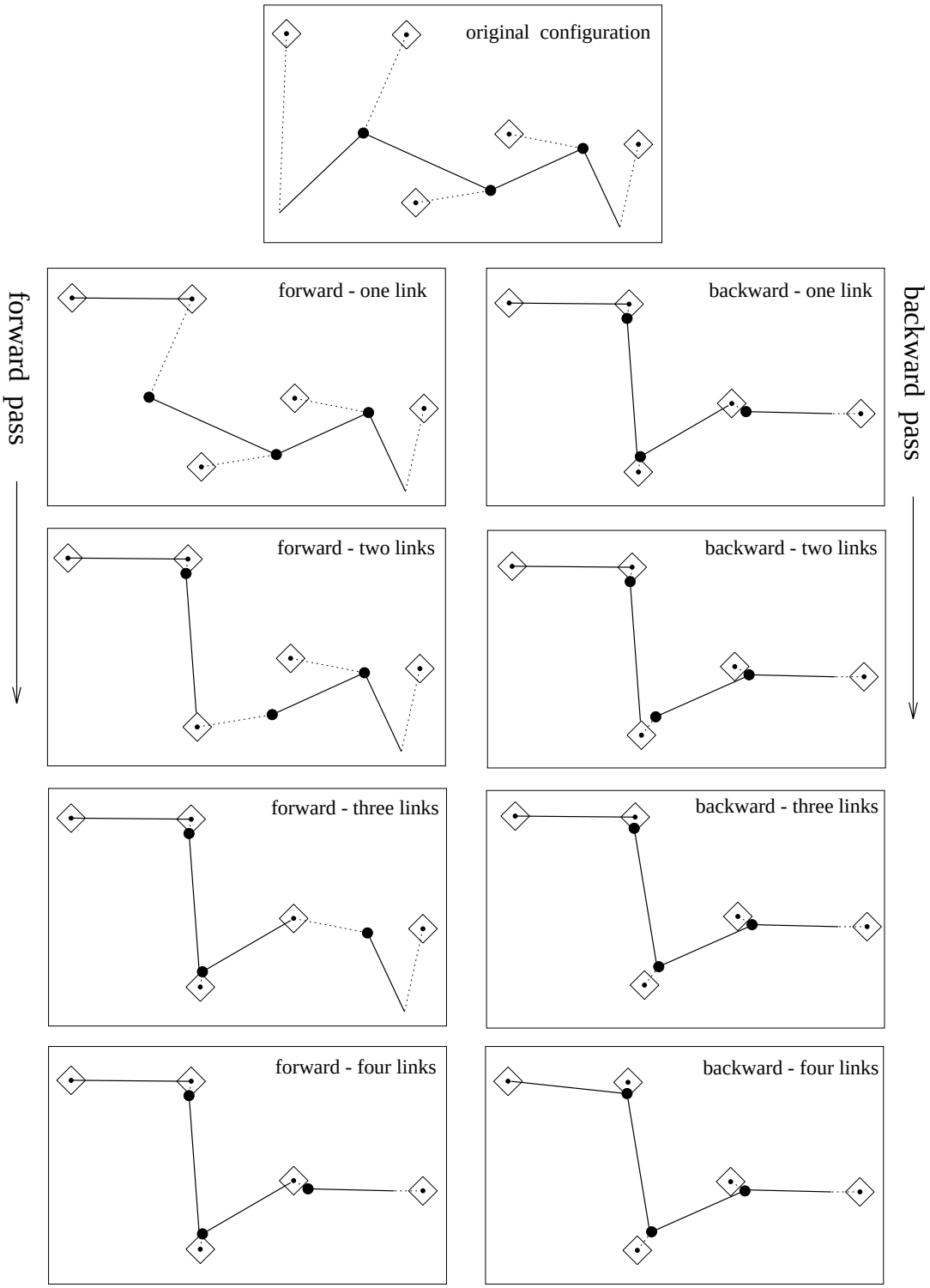

Figure 5.13: Forward pass and backward pass for a chain object. Note that the links connect to each other only during the backward pass. The lower right square is the final solution of this configuration.

5.8.2 Dynamic Solution for a Tree Structured Model

Estimating the pose of a tree structured articulated object is a little more complicated than the chain-like object. An articulated tree structured model can have more than two constraints associated with a model point, however, it does not contain a cycle of constraints (see example in Figure 5.14). Since the chain structure is a particular case of

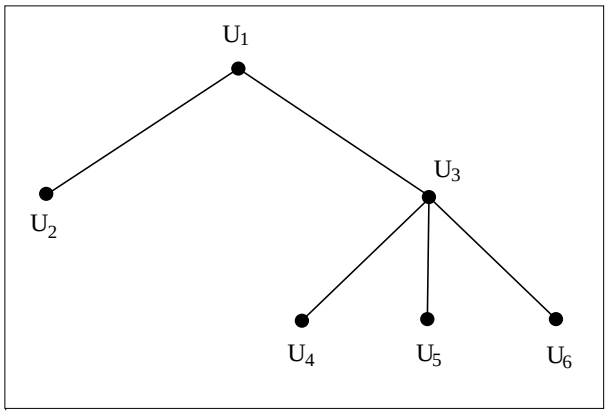

Figure 5.14: A tree structured model.

a tree structure we can easily expand the method to deal with this case. Using the same graph representation $G = (V, E)$ for a tree structured model, the algorithm solving this structure is similar to that of a chain model where the two passes are now:

1. *Bottom-up* - which passes through the sons of a node prior to the node.
2. *Top-down* - which passes through the parent of a node prior to the node.

The propagation of information is performed as follows:
Assume node k has m sons: $k_1, \ldots, k_m$ and a single parent node j (Figure 5.15). Further consider the measurements of a node k as *internal data* of that node. During the first pass, the information propagates from the leaves up to the root of the tree; at each node k, the estimates of $\mathbf{t}_k$ coming from the son nodes $\hat{\mathbf{t}}^{up}_{k,k_1} \ldots \hat{\mathbf{t}}^{up}_{k,k_m}$ are fused together ($\hat{\mathbf{t}}^{up}_{k,k_\ell}$ denotes an estimate for $\mathbf{t}_k$ produced by the son k_ℓ using the information in its sub-tree). This information is fused with the internal data of the node and an estimate $\hat{\mathbf{t}}^{up}_{j,k}$ for the parent node is evaluated and propagated up (see Figure 5.15).

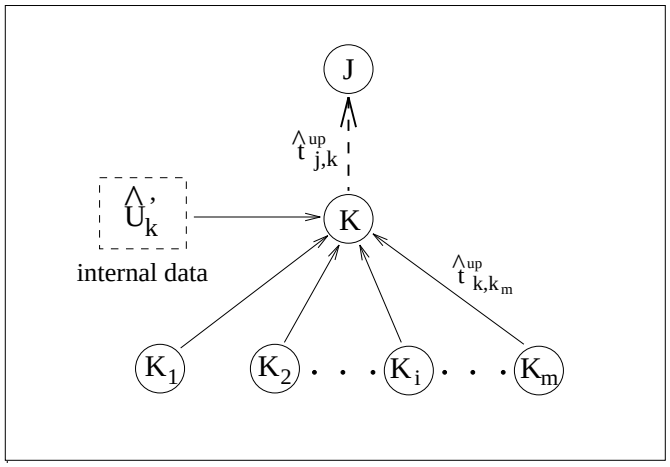

Figure 5.15: Propagation of information during the bottom-up pass: information propagates from the leaves up to the root of the tree; at each node k, the estimates of $\mathbf{t}_k$ coming from the son nodes ($\hat{\mathbf{t}}^{up}_{k,k_\ell}$, $\ell = 1 \ldots m$) are fused together with the internal data of the node and then propagated an estimate $\hat{\mathbf{t}}^{up}_{j,k}$ for the parent node is evaluated and propagated up.

In the second pass, the information is propagated from the root to the leaves of the tree, where the information obtained from the first pass is now taken into account. Node k sends an estimate $\mathbf{t}_{k_i}$ of its son's position by fusing *all* the information entering node k but *not* arriving from node k_i, i.e. the estimates $\hat{\mathbf{t}}^{down}_{k,j}, \hat{\mathbf{t}}^{up}_{k,k_1} \ldots \hat{\mathbf{t}}^{up}_{k,k_{i-1}}, \hat{\mathbf{t}}^{up}_{k,k_{i+1}} \ldots \hat{\mathbf{t}}^{up}_{k,k_m}$. This information is fused with the internal data and an estimate, $\mathbf{t}_{k_i,k}$ of the son node position is obtained (see Figure 5.16).

The propagation of information can be simply described by the following algorithm which assigns to each directed edge $(to, from)$ in the graph structure, an estimate of the state vector of node *to* as obtained from node *from*:

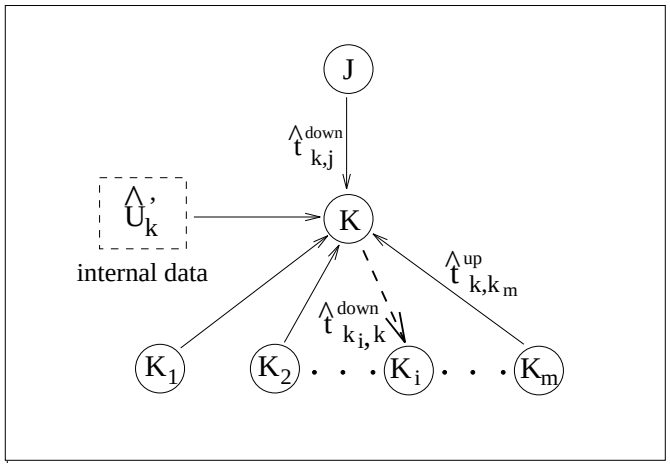

Figure 5.16: Propagation of information during the top-down pass: information propagates from the root to the leaves of the tree. Node k sends an estimate of its son node position $\mathbf{t}_{k_i}$ by fusing *all* the information entering node k but not arriving from node k_i, i.e. the estimates $\hat{\mathbf{t}}^{down}_{k,j}, \hat{\mathbf{t}}^{up}_{k,k_1} \ldots \hat{\mathbf{t}}^{up}_{k,k_{i-1}}, \hat{\mathbf{t}}^{up}_{k,k_{i+1}} \ldots \hat{\mathbf{t}}^{up}_{k,k_m}$. This information is fused with the internal data and an estimate, $\mathbf{t}_{k_i,k}$ of the son's position is obtained and propagated down.

Propagate (*from, to*):

1. Collect and fuse all the estimates that enter to node $from$ but do not come from node *to*.
2. Fuse the estimate from step 1 with the internal data (measurement) of node $from$.
3. Based on the estimate from step 2 calculate an *a priori* prediction of the position of node *to* and set it to the edge $(from, to)$.

Note that the above description adequately describes both the bottom-up pass and the top-down pass. It is used in the following general algorithm which gives an estimate for every node in the tree structure based on all the measurements and all the constraints:

The Algorithm:

1. Perform the two passes:
   (a) Bottom-up pass: traverse the graph nodes starting at the leaves and ending with the root node. For each node i which is passed, apply **propagate**(i, j) where j is the parent node of i.
   (b) Top-down pass: traverse the graph nodes starting from the root and ending at the leaves. For each node i which is passed, apply **Propagate**(j, i) where j is the parent node of i.
2. At the end of the two passes: At each node fuse the internal data of the node with all the estimates entering that node from the directional edges.

It is clear that this algorithm includes the case of a chain model which is a special case of a tree structure.

The complexity of this algorithm is $O(n)$ by fusing the information properly. This is done by first calculating, for each node k, all the estimates needed for $\mathbf{t}_k$ in a sequantial manner; We first calculate the estimates of $\mathbf{t}_k$ given the information from the son nodes k_1, $k_1..k_2$, $k_1..k_3$ etc. (as in Figures 5.15, 5.16) producing the estimates $\hat{\mathbf{t}}_{k,k_1}$,$\hat{\mathbf{t}}_{k,k_1..k_2}$, $\hat{\mathbf{t}}_{k,k_1..k_3}$ etc. respectively. In the same way we calculate all the estimates of $\mathbf{t}_k$ given the information from the nodes k_m, $k_m..k_{m-1}$, $k_m..k_{m-2}$ etc. producing the estimates $\hat{\mathbf{t}}_{k,k_m}$, $\hat{\mathbf{t}}_{k,k_m..k_{m-1}}$, $\hat{\mathbf{t}}_{k,k_m..k_{m-2}}$ etc. The time complexity of this pre-processing is linear in the number of edges in the tree and thus takes $O(n)$. After these estimates are calculated, it is obvious that each call to **Propagate** $(k,\ i)$ takes constant time since it requires at most two fusions of three entities. The routine **Propagate** is called $O(n)$ times during the two passes, therefore the complexity of the whole argorithm is $O(n)$ as stated. An example of the information propagation in a tree structure similar to the structure depicted in Figure 5.14 is given in Figure 5.17.

5.8.3 Dynamic Solution for a General Structured Model

In the previous sections we dealt with a limited subset of the constrained models, namely, those models whose constraints form a tree structure where every constraint is associated

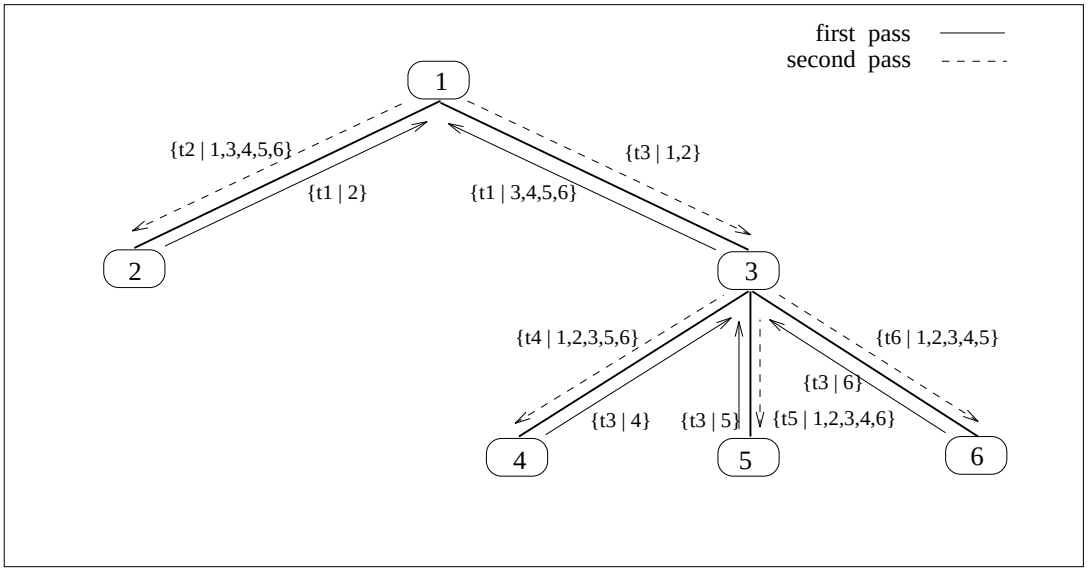

Figure 5.17: Two passes for a tree structured model. The bottom-up pass is depicted by the bold arrows and the top-down pass by the dashed arrows.

with at most two model points. Note that many real-world objects indeed fall in this category.

In this section we describe the solution for the general constrained model where the constraints may be associated with three or more model points and where the constraints form any structure which may include cycles. The general case described here, also includes those models where each component consists of more than one point (since the rigidity of the multi-point-component can be enforced by a set of constraints on the points of the component). However, we will deal with these cases separately, in Section 5.9.

Denote by ψ_k a model constraint and by $Point(\psi_k)$ the model points associated with this constraint. In order to estimate the pose of the model points, we build an *estimation-graph* $\Gamma = (\Omega, \Xi)$ which is defined as follows: Every vertex of the graph $\omega_i \in \Omega$ represents a set of model points such that Ω is a partition of the model points. Every edge $\xi_{ij} \in \Xi$ in the estimation-graph represents a connection between vertices ω_i and ω_j. Every model constraint is associated either with a vertex or with an edge of the estimation-graph; if $Point(\psi_k) \subseteq \omega_i$ then ψ_k is associated with the vertex ω_i and is defined as an *internal constraint*. In this case, $\psi_k \in Const(\omega_i)$ where $Const(\omega_i)$ is the set of internal con-

straints associated with vertex ω_i. If $Point(\psi_k) \subseteq (\omega_i \cup \omega_j)$ then ψ_k is associated with the edge ξ_{ij} and is defined as a *mutual constraint*. The set of all mutual constraints between vertices ω_i and ω_j is denoted $Const(\xi_{ij})$. An example of a constrained model and its estimation graph is presented in Figure 5.18. In this example ψ_1 and ψ_4 are internal constraints and ψ_2, ψ_3 are mutual constraints.

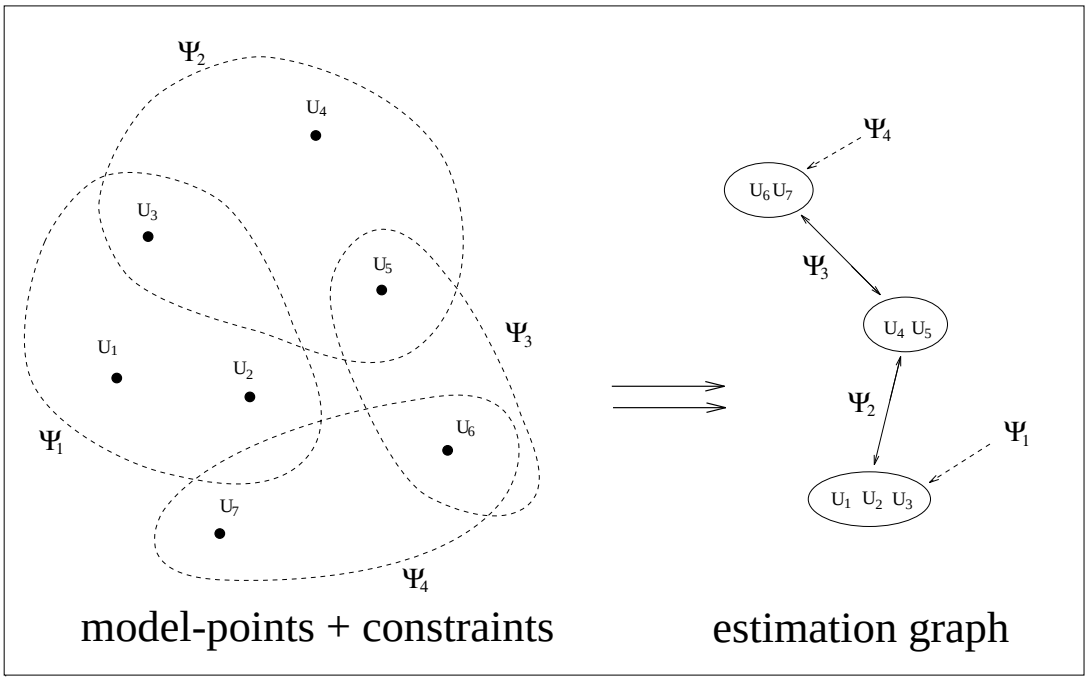

Figure 5.18: An example of a constrained model and its estimation graph. ψ_1 and ψ_4 are internal constraints and ψ_2, ψ_3 are mutual constraints. z

The construction of the estimation-graph will enforce each constraint to be defined as either an internal constraint or a mutual constraint and no constraint will be associated with more than two vertices (i.e. $\forall \psi_k \; \exists \omega_i, \omega_j$ s.t. $Point(\psi_k) \subseteq (\omega_i \cup \omega_j)$). For simplicity, we describe the construction of the estimation graph as a two stage process: The first stage defines the vertices of the graph and the second stage introduces the constraints. The division into two stages is for didactic purposes, in practice, the graph is constructed in a single stage.

Construction of the Estimation-Graph

First Stage - Defining the vertices of the graph:
This stage requires two passes over the constraints of the model. In the first pass an initial definition of the vertices is constructed as follows:

1. $\Omega = \emptyset$; $V = \emptyset$
2. **For each** constraint ψ_i **do**
   (a) $\omega = Point(\psi_i) - V$
   (b) **if** $\omega \neq \emptyset$ **then**
      i. add ω as a new vertex of Ω
      ii. $V = V \cup \omega$

The second pass merges vertices so that no constraint will depend on more than two vertices of the graph (this pass can use algorithms for construction of equivalence classes [1]):

1. **For each** constraint ψ_i **do**
   (a) **if** there is more than two vertices: $\omega_{i_1}, \omega_{i_2}, \cdots, \omega_{i_k}$
   s.t. $Point(\psi_i) \cap \omega_{i_j} \neq \emptyset$ **then**
   exchange $\omega_{i_1}, \omega_{i_2}, \cdots, \omega_{i_k}$ with a new vertex ω_{new} where
   $\omega_{new} = point(\omega_{i_1}) \cup point(\omega_{i_2}) \cup \cdots point(\omega_{i_k})$.

At the end of this pass, no model constraint is associated with more than two graph vertices. The described two passes are a simple suggestion for the arrangement of the model points in the vertices graph. This scheme is not unique and some heuristics can be used in order to enhance the arrangement (for example, some vertices can be merged to ω_i instead to ω_{new} so that ω_{new} and ω_i will contain about the same number of points).

Second Stage - Introducing the constraints into the graph:

At this stage the constraints are associated with their graph vertices/edges.

1. $\Xi = \emptyset$
2. **For each** constraint ψ_i **do**
   (a) **if** $\exists \omega_j$ s.t. $Point(\psi_i) \subseteq \omega_j$ **then** add ψ_i to $const(\omega_j)$
   (b) **else**
      i. find ω_k, ω_j s.t. $Point(\psi_i) \subseteq (\omega_k \cup \omega_j)$
      ii. add ξ_{kj} to Ξ
      iii. add ψ_i to $const(\xi_{kj})$

since the number of constraints is less than $3n$ (as discussed in Section 5.6.1), the complexity of constructing the estimation-graph is almost linear in n, the same as the complexity of the equivalence classes algorithm [1].

Evaluation of the Estimation-Graph

In the case where the estimation-graph forms a tree, the evaluation is similar to that described in Section 5.8.2. Every vertex ω_k of the graph is considered a single point in $3 * \|\omega_k\|$ dimensional space and the state-vector associated with it is:

$$\begin{pmatrix} \mathbf{t}_k^1 \\ \mathbf{t}_k^2 \\ \vdots \end{pmatrix} = \tilde{\mathbf{t}}_k$$

where $\mathbf{t}_k^i$ is the position vector of a model point in ω_k. The internal-data of ω_k is the internal estimate of $\tilde{t}_k$. This estimate will be evaluated from the measurements associated with $\tilde{t}_k$ and from the internal constraints of ω_k. The internal estimate is calculated using the iterative method similar to that described in Section 5.7. The resulting internal-estimation is denoted $(\tilde{t}_k^{int}, \Sigma_k^{int})$. Following this initial stage, the estimation of all state-vectors in the graph is evaluated according to the algorithm for a tree-structured model (as described in Section 5.8.2).

In the case where the estimation-graph contains cycles of constraints, one of the following two strategies can be followed:

1. Breaking the constraint cycle:
   W.l.o.g. assume that $\psi_{k-1}(\tilde{\mathbf{t}}_{k-1}, \tilde{\mathbf{t}}_k)$ is a mutual constraint that is part of a constraint cycle and is the only one in $Const(\xi_{k-1,k})$. The information supplied by ψ_{k-1} can be fused with the estimates of $\tilde{\mathbf{t}}_k$ and of $\tilde{\mathbf{t}}_{k-1}$ using a method similar to that described for the predictor (Section 5.8.1) as follows:
   The state-vector is taken as:
   $$\begin{pmatrix} \tilde{\mathbf{t}}_{k-1} \\ \tilde{\mathbf{t}}_k \end{pmatrix}$$
   and the *a priori* estimate input is taken as:
   $$\left[\begin{pmatrix} \tilde{\mathbf{t}}^{int}_{k-1} \\ \tilde{\mathbf{t}}^{int}_{k} \end{pmatrix}, \begin{pmatrix} \Sigma^{int}_{k-1} & 0 \\ 0 & \Sigma^{int}_{k} \end{pmatrix} \right]$$
   where $(\tilde{\mathbf{t}}^{int}_{k-1}, \Sigma^{int}_{k-1})$ and $(\tilde{t}^{int}_{k}, \Sigma^{int}_{k})$ are the internal estimates of ω_{k-1} and ω_k respectively.
   The measurement model input is equivalent to that given in Eq.5.6 where $\mathbf{m}_k$ will be considered as the measurement input with zero uncertainty.

   The result obtained from the K.F. fuser is an estimate of $\tilde{\mathbf{t}}_{k-1}$ and $\tilde{\mathbf{t}}_k$ based on the internal estimates $\tilde{\mathbf{t}}^{int}_{k-1}$ and $\tilde{\mathbf{t}}^{int}_{k}$ of vertices ω_{k-1} and ω_k and based on the constraints between them. In the rest of the process, these estimates will be taken instead of the internal estimates of ω_{k-1} and ω_k. After fusing the constraint ψ_{k-1} we disconnect the graph edge $\xi_{k-1,k}$ associated with this constraint. We follow this strategy until all cycles of constraints are broken and we obtain a tree structured graph. Following this stage we continue the estimation as elaborated for a tree structured model. Using this strategy, the "measurement" ψ_{k-1} is considered twice; through the information flowing from $\tilde{\mathbf{t}}_{k-1}$ and through the information flowing from $\tilde{\mathbf{t}}_k$. However, considering the constraint "measurement" twice will not affect the final solution since the uncertainty of this "measurement" is zero and its weight relative to the other measurements is, in any case, infinite.

2. Reduction of a cycle to a single node:
   In this method, all vertices in the cycle are combined into a single vertex. The

method of combining these vertices is similar to the method described for building the estimation-graph where we consider the constraint-cycle as a single constraint associated with all vertices in the cycle. All cycles in the graph are similarly reduced until a tree-structure is obtained. An example of this reduction process is shown in Figure 5.19. This process may result in a reduction of the estimation-graph into a single vertex, in which case the dynamic method of pose estimation is reduced to the batch method.

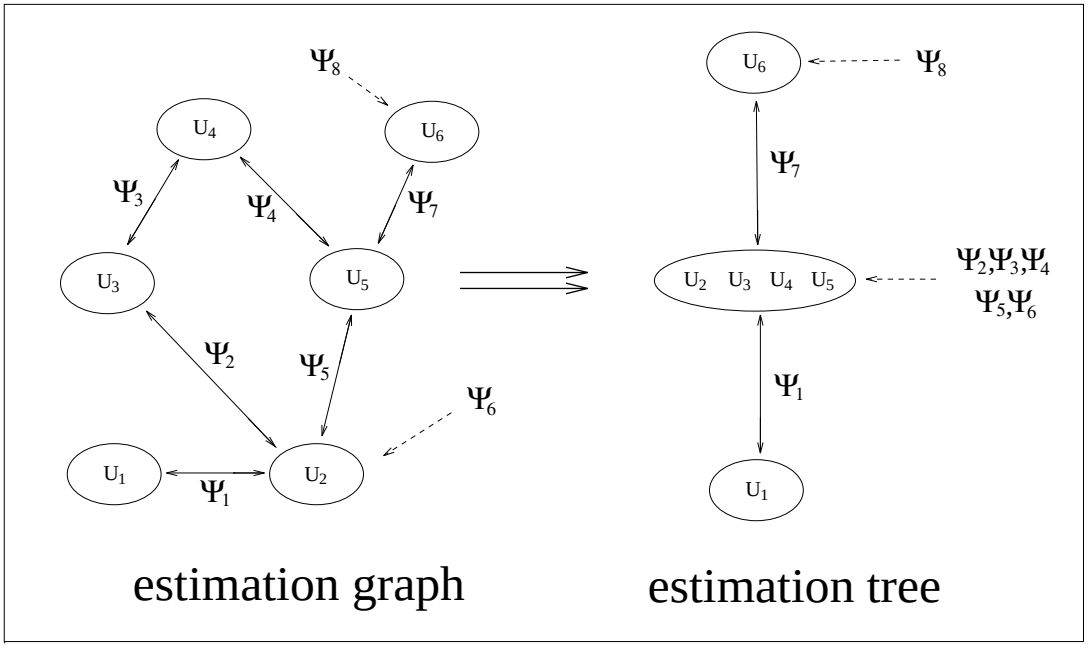

Figure 5.19: Reducing an estimation graph and obtaining an estimation tree. Every constraint-cycle in the graph is considered as a single constraint associated with all vertices in the cycle. Thus U_2, U_3, U_4, U_5 are merged into a single vertex and the constraints $\psi_2, \psi_3, \psi_4, \psi_5$ are associated with it.

Breaking all the cycles in a graph is done by calculating a spanning tree of the graph and deleting all the extreneous edges as described above. Since the number of constraints does not exceed $3n$ (where n is the number of model points), the initial estimation graph has no more than $3n$ edges. Thus, the complexity of any one of the two methods is $O(n \log n)$ which is the complexity of finding a spanning tree in the graph.

5.8.4 The Dynamic Approach: Computational Aspects

The complexity of a single global iteration of the dynamic method, depends on the structure of the estimation-tree. In the extreme case where the tree forms a chain-structure and every node represents a single model point, the complexity is $O(n)$. In the other extreme case, where the estimation-tree is reduced to a single node, the complexity is similar to that of the batch method and is $O(n^3)$ (though this case is quite rare).
The complexity of constructing the estimation-graph and reducing the constraint cycles, is of order $O(n \log n)$ and so, generally, does not impose heavily on the overall computation time. Moreover, the estimation-tree can be constructed off-line, i.e. once for each model, prior to the beginning of the estimation process.

Thus, in most cases, the dynamic solution requires less computation time than other methods and is more economic both in the number of global iterations and in the number of operations in each iteration. We note that the improvement in the complexity is due to a better exploitation of the sparsity of the matrix H (Eq. 5.4). As the matrix H becomes less sparse, the complexity of the dynamic solution increases and approaches that of the batch method. More precisely, as the estimation-graph has more nodes, and as the valency of the graph decreases, the complexity of the process approaches $O(n)$.

5.8.5 The Measurement Interpretation

As in the iterative method, here too, the interpretation process and the estimation process can be performed simultaneously.

In the first pass (bottom-up), the *a priori* estimate obtained for the state-vector of node ω_k, takes into account all measurements and constraints associated with the subtree of ω_k. This estimate is used to eliminate irrelevant matches of the model points in ω_k. The rejection and acceptance of a match is performed as described for the iterative method, however the state vector is taken as $\tilde{\mathbf{t}}_k$ rather than T.

In the second pass (top-down), following the fusion of all information for the estimation of $\tilde{\mathbf{t}}_k$ and prior to the estimation of the ω_k-subtree nodes, we perform a goodness-of-fit test between the matched measurements and the final estimates of the model points of ω_k. This test will serve as the verification process.

Thus, the interpretation process is performed during the first pass (bottom-up) and the verification process is performed during the second pass (top-down).

5.9 Constrained Objects Having Multiple-Point Components

We easily extend the solution for objects having one point per component to objects that have multiple-point components. For every component C_k, one must estimate the transformation $\hat{\mathbf{T}}_k$ which describes the pose of the component. The transformation $\mathbf{T}_k$ is composed of a quaternion $\tilde{q}_k$ and a translation vector $\mathbf{t}_k$:

The process of evaluating all the transformations $\{\mathbf{T}_i\}$ is similar to the three methods previously described for models having a single point per component as following:

1. Using the batch paradigm - The solution consists of two computational phases. In the first phase the transformation $\hat{\mathbf{T}}_k$ is estimated for each component C_k using the measurements associated with the points belonging to this component. In this phase the constraints existing between the components are not taken into consideration and the position of each component is estimated as if it was the whole object. The pose estimation of each component is computed as explained in Chapter 4. At the end of the first phase we have a set of evaluated transformations and their uncertainties:

$$\hat{\mathbf{T}} = \begin{pmatrix} \hat{\mathbf{T}}_1 \\ \hat{\mathbf{T}}_2 \\ \vdots \\ \hat{\mathbf{T}}_n \end{pmatrix} \quad \text{and} \quad \Sigma = \begin{pmatrix} \Sigma_1 \\ \Sigma_2 \\ \vdots \\ \Sigma_n \end{pmatrix}$$

   In the second phase we consider $(\hat{\mathbf{T}}, \Sigma)$ as an *a priori* estimation for the transformations and we consider the set of constraints as the artificial "measurements". Fusing the constraint "measurements" with the *a priori* estimations is done as in the single-point component case.

2. Using the incremental paradigm - The state-vector in this method is T where the information obtained from a measurement is fused as described in Chapter 4 for a

rigid object, and the information obtained from a constraint is fused as described in Section 5.7. The order in which the constraints are fused, is obtained using the algorithm given in Section 5.7.1 where the components $\{C_k\}$ take the place of the model points. The order described by the algorithm, ensures that prior to fusion of a point in any component, all available information from neighboring components and mutual constraints have been exploited in order to assist in rejecting irrelevant matches.

3. Using the dynamic method - In this method the estimation-graph is given again as a graph $\Gamma = (\Omega, \Xi)$ where each node represents a set of components (rather than model points) and every edge $\xi_{i,j} \in \Xi$ represents a constraint between components in ω_i and components in ω_j. For each node ω_k, the state-vector to be estimated is a concatenation of the transformations of all components in ω_k:

$$\tilde{\mathbf{T}}_k = \begin{pmatrix} \mathbf{T}_k^1 \\ \mathbf{T}_k^2 \\ \vdots \end{pmatrix}$$

The algorithm and the order of fusion of the constraints and measurements, follow the description given for the dynamic method (Section 5.8.3) but, in this case, components are taken instead of points. The fusion of internal information coming from measurements is the fusion of all measurements associated with the components of the node. This fusion is performed as described for rigid objects (Chapter 4).

In practice, most constrained objects are composed of a small number of components, each with a relatively large number of model points and each constraint depends on only two components. The estimation-graph of such objects will have nodes representing a single model component. The dynamic method is most appropriate and efficient for evaluating the pose of these models.

Further, in the case where every component contains several model points, there is no need to restrict the measurements to 3D since the pose of the component can be estimated from projections (2D or 1D) as described in Chapter 4.

5.10 Inequality Constraints

Inequality constraints appear in pose estimation mainly in context of recognition. These constraints assist in rejection of inconsistent interpretations that contradict the inequalities. Examples of such inequality constraints can be found in articulated models such as scissors and robot arms that are limited in the range of feasible angles between parts. Another example can be found in the work by Grimson [26, 27] where points are restricted to match segments of the model by limited the range of distances between points.

The inequality constraints can be reduced to equality constraints by rewriting a given constraint:

$$g(x) \geq 0$$

as an equality constraint

$$g(x) - \lambda^2 = 0$$

where λ is a new variable that is added to the state vector and is estimated during the filtering process. Thus, every inequality constraint increases by one the dimensionality of the vector to be estimated. The initial *a priori* uncertainty associated with the parameter λ is infinite.

5.11 Results

5.11.1 Simulated Data

We applied our method to estimate the pose of a $2D$ constrained model consisting of single point components. We used the parametric modeler described in [31] which we developed based on our techniques. The modeler enables the definition of constraints graphs using the following types of constraints: co-linearity of three points, a particular distance between two points, a particular distance between a point and a fixed location, constraining a point to lie on a fixed line and constraining a point to be on one side of a fixed line (inequality constraint). It was demonstrated that the algorithm is capable

of computing solutions to complex models. Not surprisingly, all three of the presented techniques gave the same solution for different model configurations.

The following figures were created using this software. In the figures, model points and constraints are represented as shown in Figure 5.20. Figure 5.21 shows a tree struc-

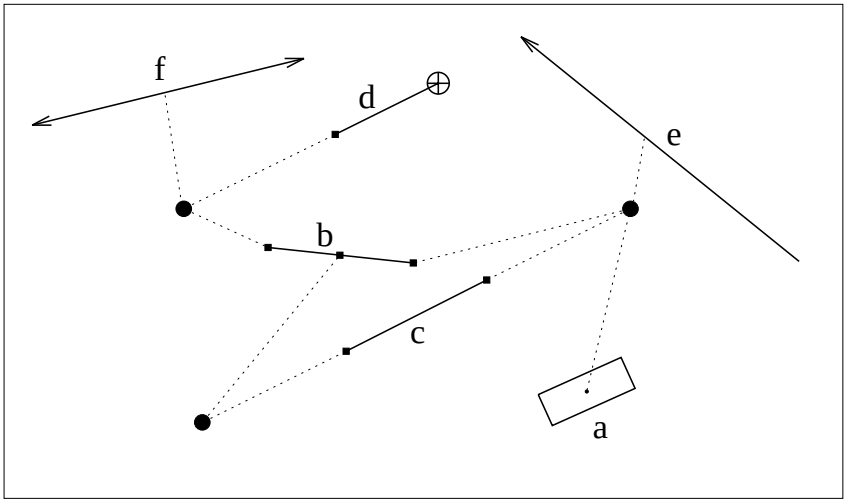

Figure 5.20: An example of measurements and constraints on three model points. The model points to be positioned are shown as full circles. A measurement (a) is described as a rectangle positioned at the measurement location and having width and length proportional to the s.t.d. of the measurement. A *fixed location distance* constraint (d) is visualized by a line segment with one endpoint at the fixed location and the other connected to the constrained point. An *inter-point distance* constraint (c) is visualized by a line segment connecting the constrained points. A *three points co-linearity* constraint (b) is shown by a fixed length line segment connected on both its ends and its middle to the constrained points. A *point on right of line* constraint (e) is shown by an arrow headed line segment connected to a model point. A *point on line* constraint (f) is shown by a double headed arrow line connected to a model point.
The connections between constraints and associated model points, are marked by dashed lines.

ture model having 6 distance constraints. These figures display the resulting pose of the model, given different measurements. As can be seen, the model constraints are conserved independent of the measurement location and variance. Figure 5.22 shows two examples

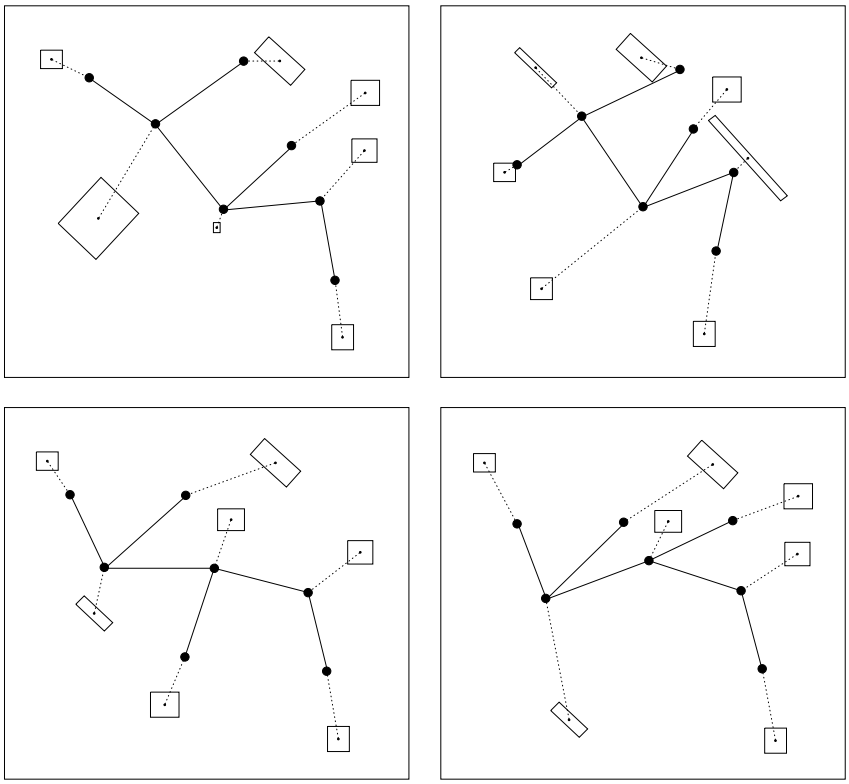

Figure 5.21: A tree structured model having 6 distance constraints. These figures display the resulting pose of the model, given different measurements. As can be seen, the model constraints are conserved independent of the measurement location and variance.

of constrained models having a general graph structure (including a circle of constraints) with their solutions. An example of a system containing inequality constraints is shown in Figure 5.23. In these cases model points are assumed to be on the right of a certain arrow headed line. Figure 5.24 shows more complex examples which include inequality constraints.

In order to confirm the validity of the results, we applied the suggested methods to several simple examples containing only two degrees of freedom. The obtained results were compared to energy maps describing the energy of the analogous physical systems. Figures 5.25 and 5.26 show two examples of constrained objects having two degrees of freedom denoted θ and ϕ. The energy maps corresponding to these examples describe

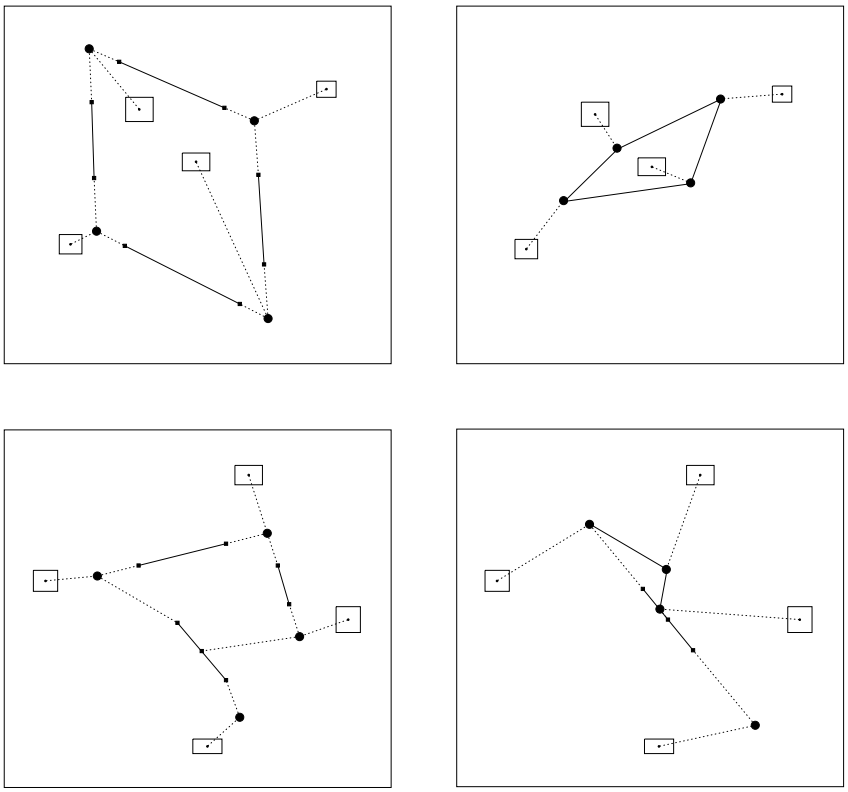

Figure 5.22: Two examples of constrained models having a circle of constraints. The initial guess (left) and the final solution (right).

the energy of the system for every θ, ϕ pair (Figures 5.25 and 5.26 at the bottom). The displays show that the solution obtained by the suggested method (marked as X in the energy map) indeed corresponds to the minimum energy solution.

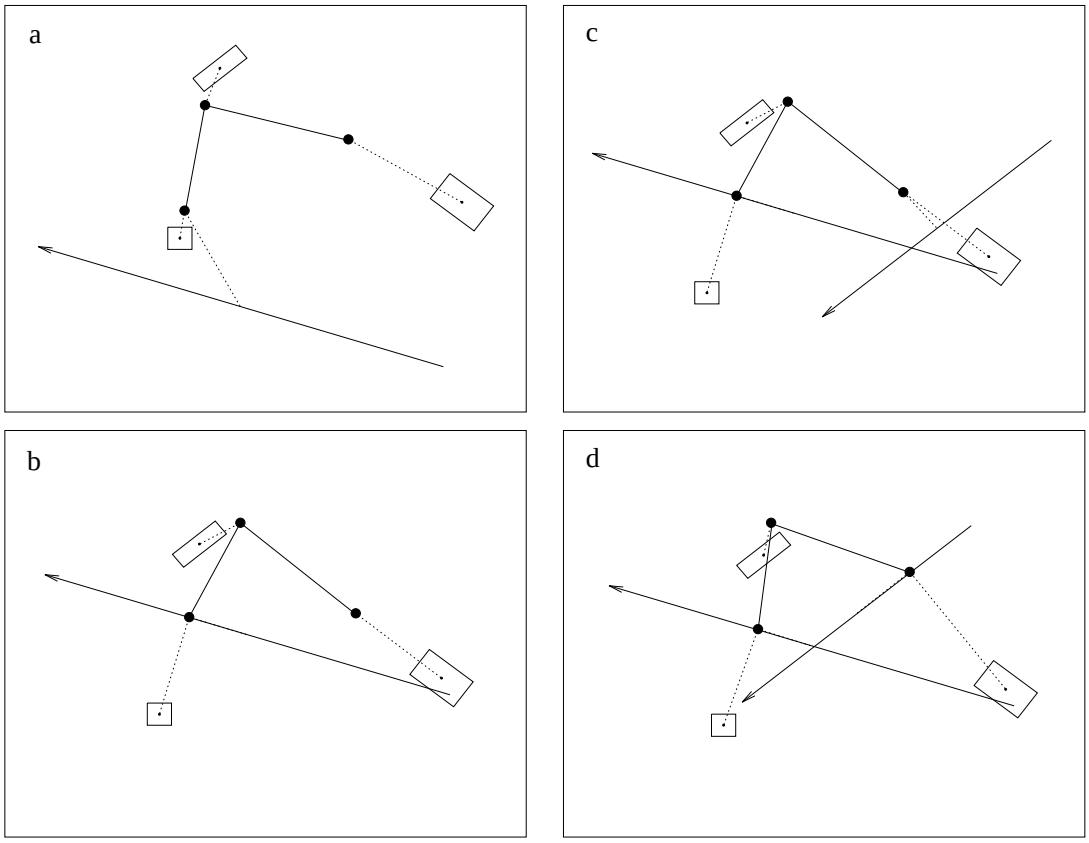

Figure 5.23: Four examples of an object having the same measurement configuration with different inequality constraints. A model point that is connected to an arrow-headed line (with a dashed line) is constrained to be to the right of this arrow. It can be seen that the arrow constraint "pushes" the model point upward in Figures b and d. When the inequality constraint is not in conflict with the apriori solution, the final solution is not influenced by this constraint (Figures a and c).

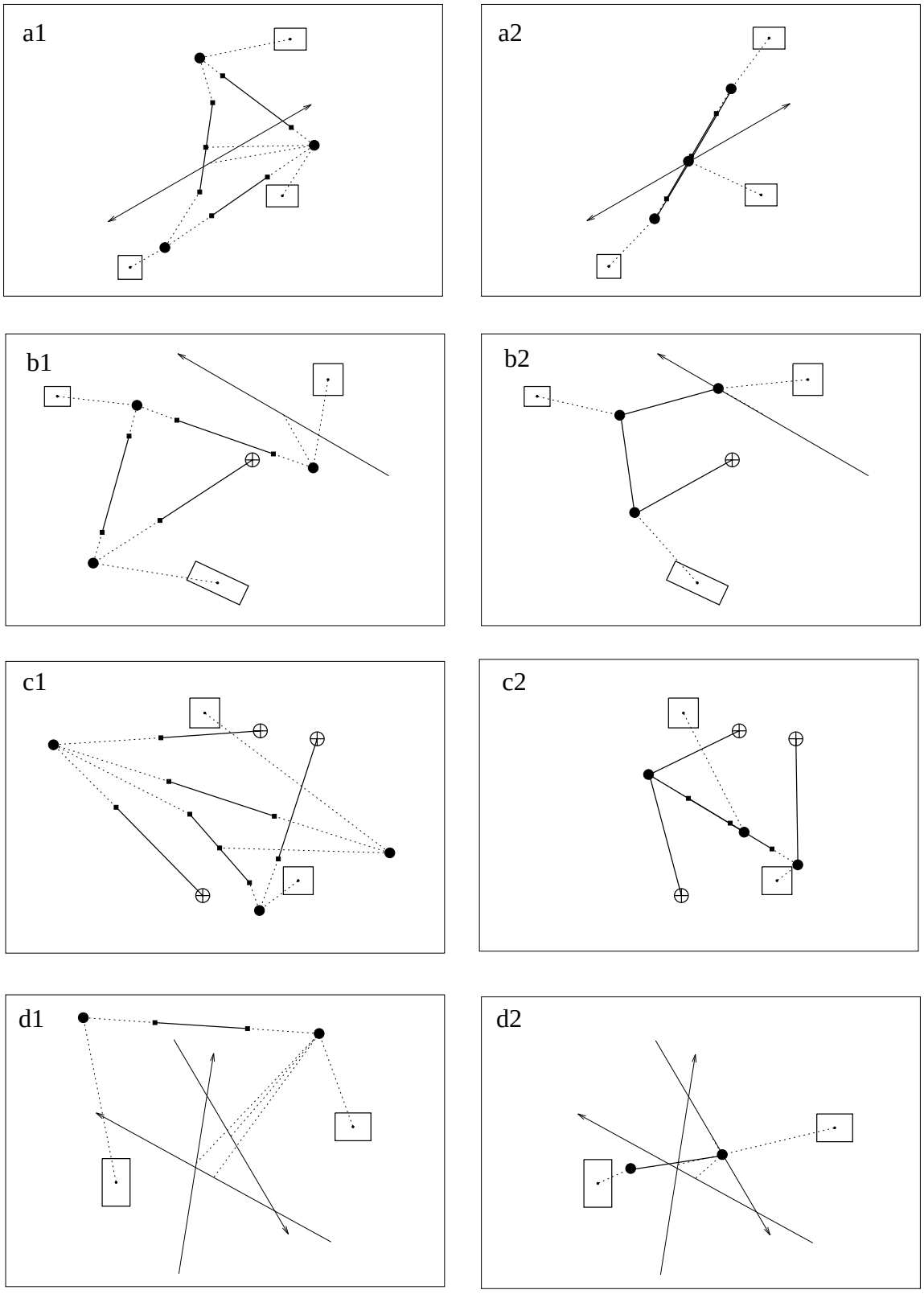

Figure 5.24: Some examples of articulated constrained models including inequality constraints. Four examples are shown, the original input (measurements, constraints and model points) are shown on the left and the solution is shown on the right.

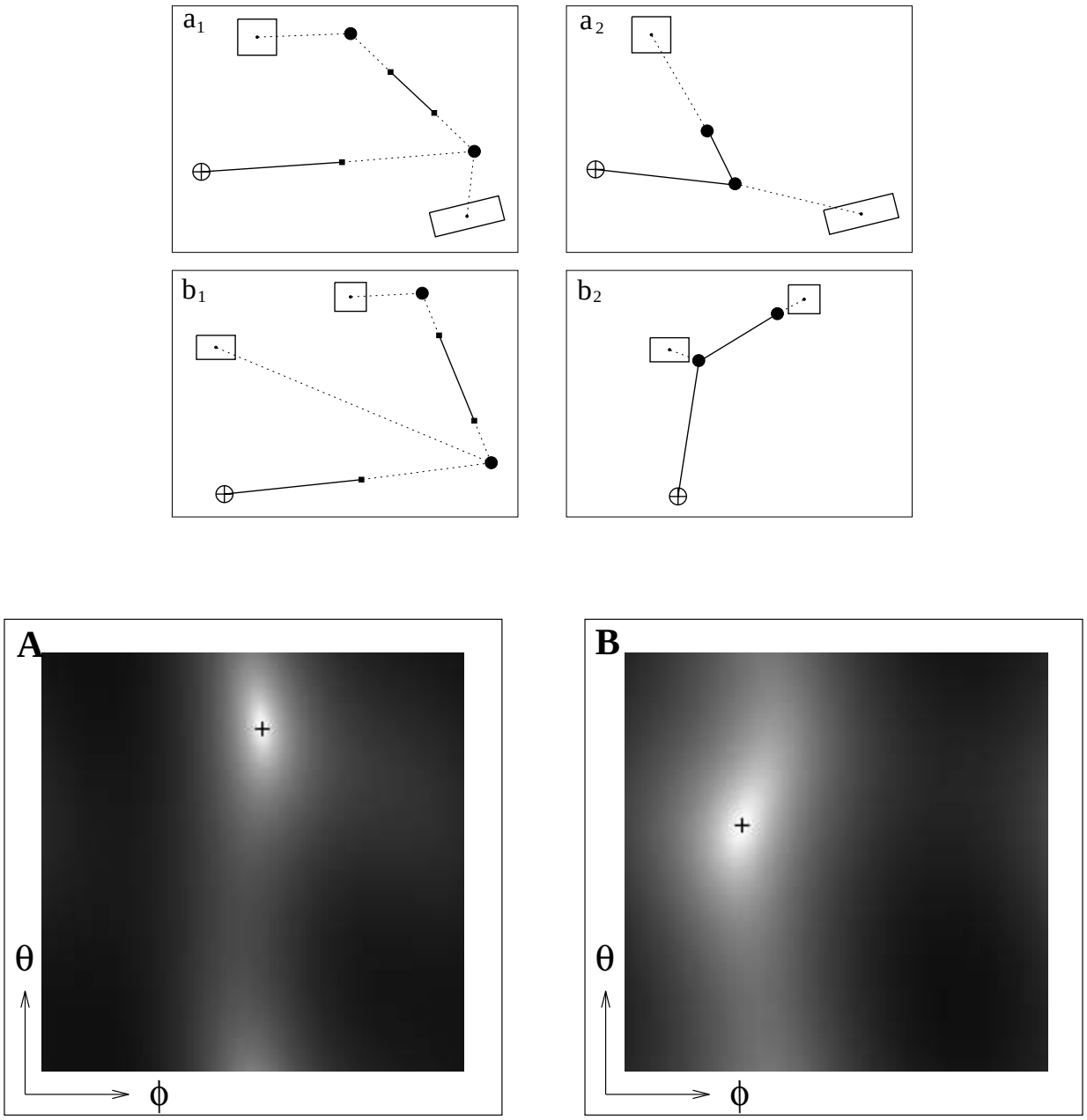

Figure 5.25: Two cases (a and b) of a constrained model with two degrees of freedom (denoted θ and ϕ). a_1, b_1 - the initial guess of the solution. a_2, b_2 - the final solution. The energy maps corresponding to each case is shown at the bottom. White values denote low energy of the system. The black crosses mark the θ, ϕ corresponding to the final solution of a_2, b_2. It is seen that the final solution is indeed correspond to the minimum energy.

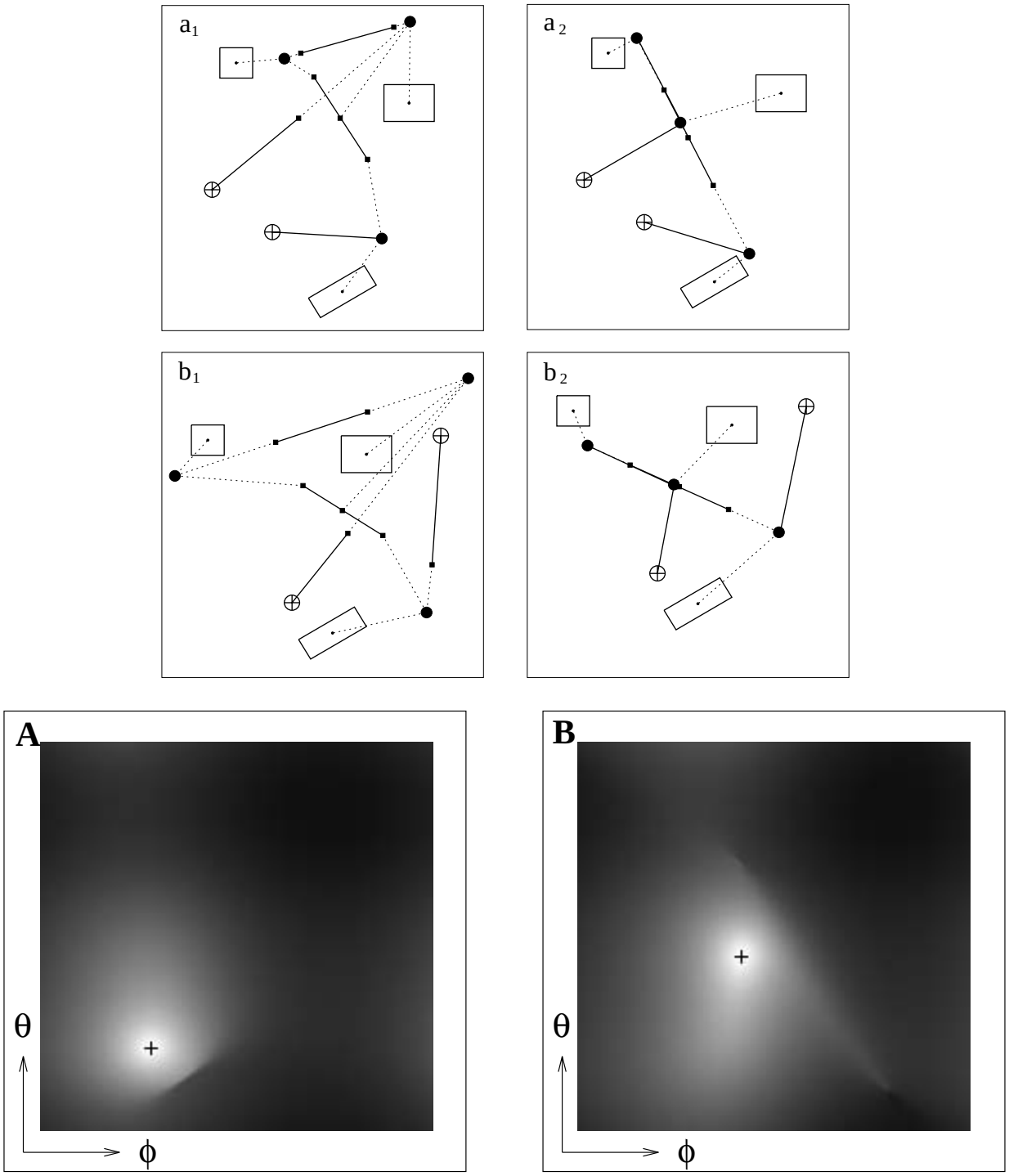

Figure 5.26: Two cases (a and b) of a constrained model with two degrees of freedom (denoted θ and ϕ). a_1, b_1 - the initial guess of the solution. a_2, b_2 - the final solution. The energy maps corresponding to each case is shown at the bottom. White values denote low energy of the system. The black crosses mark the θ, ϕ corresponding to the final solution of a_2, b_2. It is seen that the final solution is indeed correspond to the minimum energy.

5.11.2 Real Image Data

We applied our method to estimate the position of a real articulated 3D object from 2D images. The articulated model used, is a desk lamp shown in Figure 5.28, having 5 degrees of freedom. We consider the lamp model as a 23-point model (as shown in Figure 5.27) and we included the following constraints into the model:

- constant distance constraints between couples of points in the model (for example points 10 and 12).
- parallel constraints between 2 pairs of points (between points 6 and 16 and points 10 and 15).
- co-planar constraints between 4 or more points (points 10,12,13 and 9 are constrained to be co-planar).

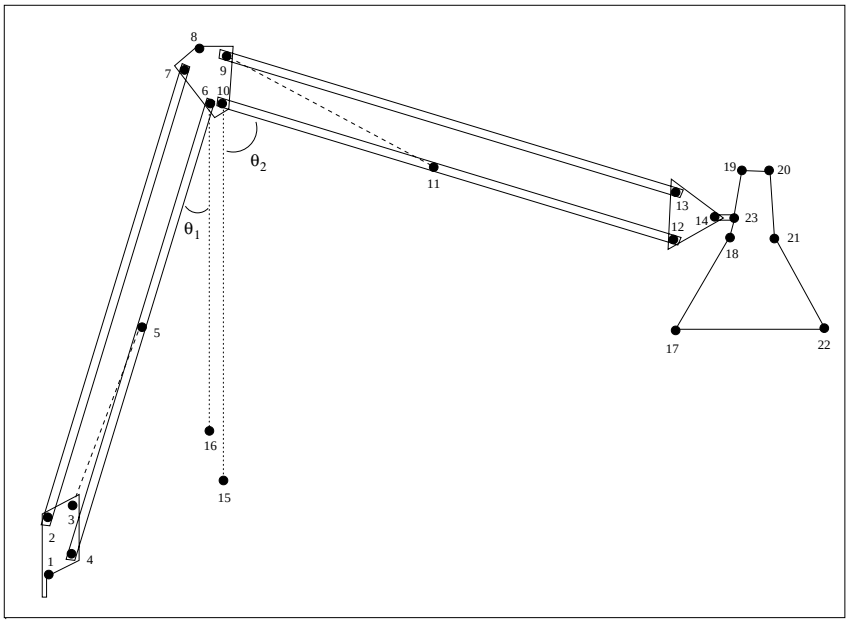

Figure 5.27: A schematic diagram of a lamp model having 23 points.

Measurements of the 3D location of the points and the measurement uncertainty were obtained from stereo image pairs. This data is noisy due to digitization, inconsistent

lighting and imprecise feature matching. The uncertainty due to noise were modeled according to the auto-correlation of the image features [68]. We estimated the pose of the lamp components from the noisy 3D measurements and the constraints using our technique. The evaluated vector is a $23 \cdot 3$ dimensional location vector composed of the 23 locations of the model points. Figures 5.28a and 5.28b show 2 examples of lamp images having 2 different positions. Figures 5.28c and 5.28d show the corresponding results as synthetic images created from the estimated location vector. As can be seen, there is high correlation between the real model location and the synthesized reconstruction.

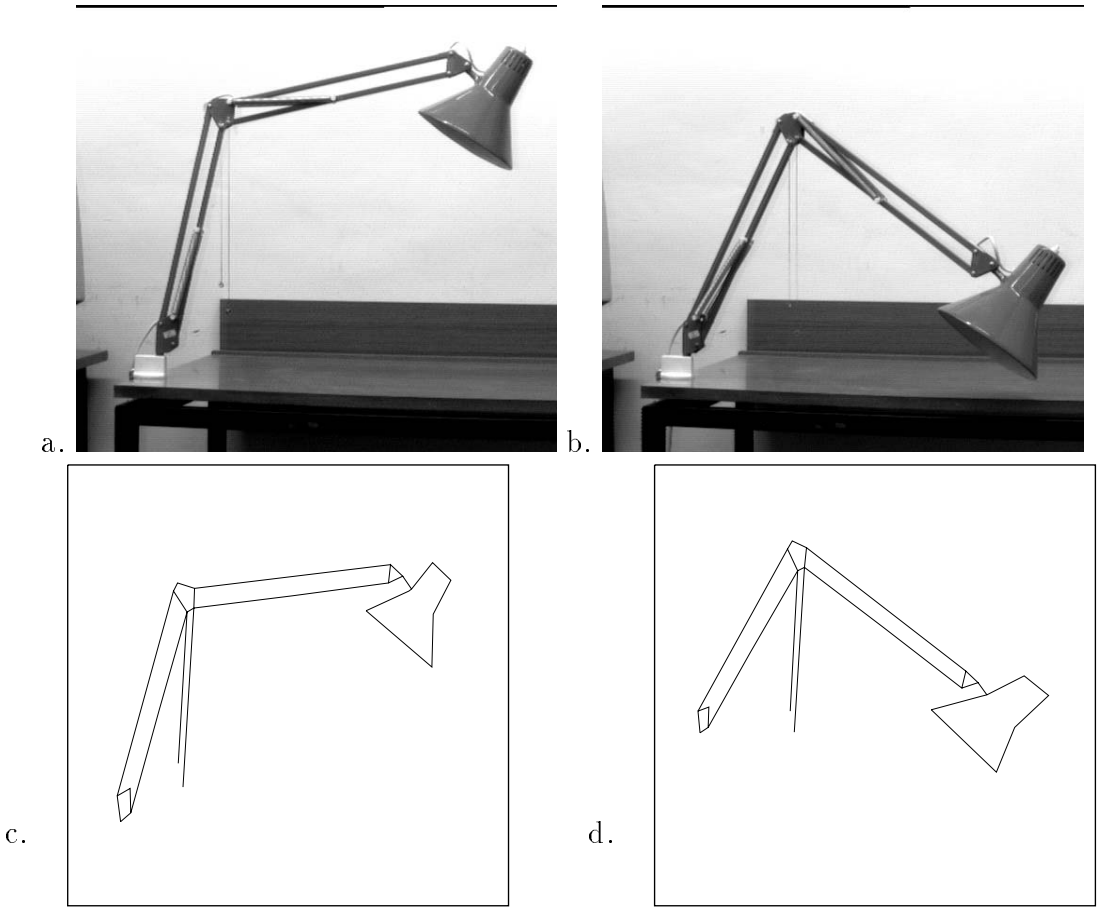

Figure 5.28: a-b) Images of a desk lamp at different positions. c-d) The corresponding result shown as a synthetic image created from the estimated location vector.

Additionally, the angles θ_1 and θ_2 (shown in Figure 5.27) were physically measured in several positions of the lamp. These values were also extracted from the pose estimate obtained with our method. The real values and the constructed values for four typical examples are compared in the following table:

		pose A	pose B	pose C	pose D
Real Values	θ_1	17.0	12.0	14.0	26.0
	θ_2	61.0	103.0	84.0	56.0
Reconstructed Values Without Constraints	θ_1	20.2	13.6	13.7	29.3
	θ_2	58.4	104.6	80.2	53.4
Reconstructed Values With Constraints	θ_1	16.5	12.3	14.1	25.8
	θ_2	60.8	104.2	84.7	57.4

The table values show the improvement in the reconstructed angle values when the fusion includes the constraints. The difference between the real angle values and the reconstructed values decreases when fusing the constraints. The s.t.d of these differences is 2.60 for the reconstruction without fusing the constraints and 0.73 for the reconstruction with constraint fusion.

The importance of propagating the pose information of each component to its neighbor component, is shown in Figures 5.29 and 5.30. Figure 5.29 shows several views of a synthesized lamp reconstructed only from the 3D measurements taken on the lamp shown in Figure 5.28b. Figure 5.30 shows the same views after mutual information was propagated between the components through the constraints. The improvement is significant, as demonstrated.

Figure 5.31 shows a limited part of the interpretation tree (I.T.) which is constructed for the desk lamp interpretation. This I.T. is used for the matching process as described in Section 5.8.5. Each node on the kth level of this I.T. represents a possible matching between the kth model point, as numbered in Figure 5.27, and some particular measured point. The measurements are numbered according to their real correspondence (i.e. the true match of the kth measured point in the kth model point). The score of each match is

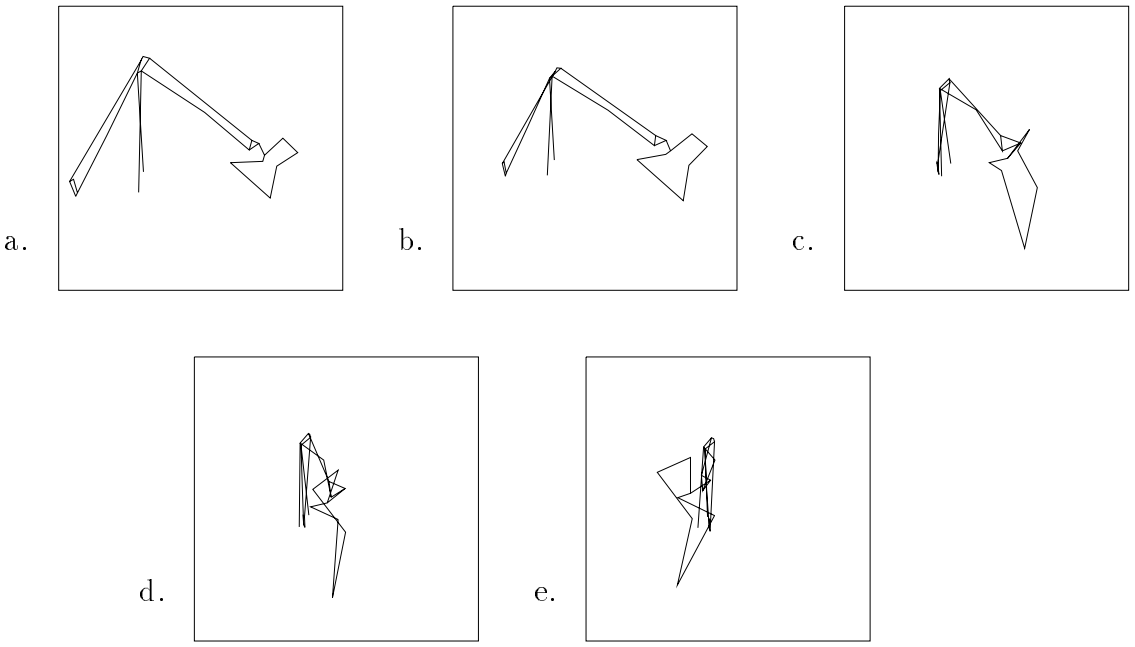

Figure 5.29: Several views of a synthesized lamp reconstructed from the 3D measurements only.

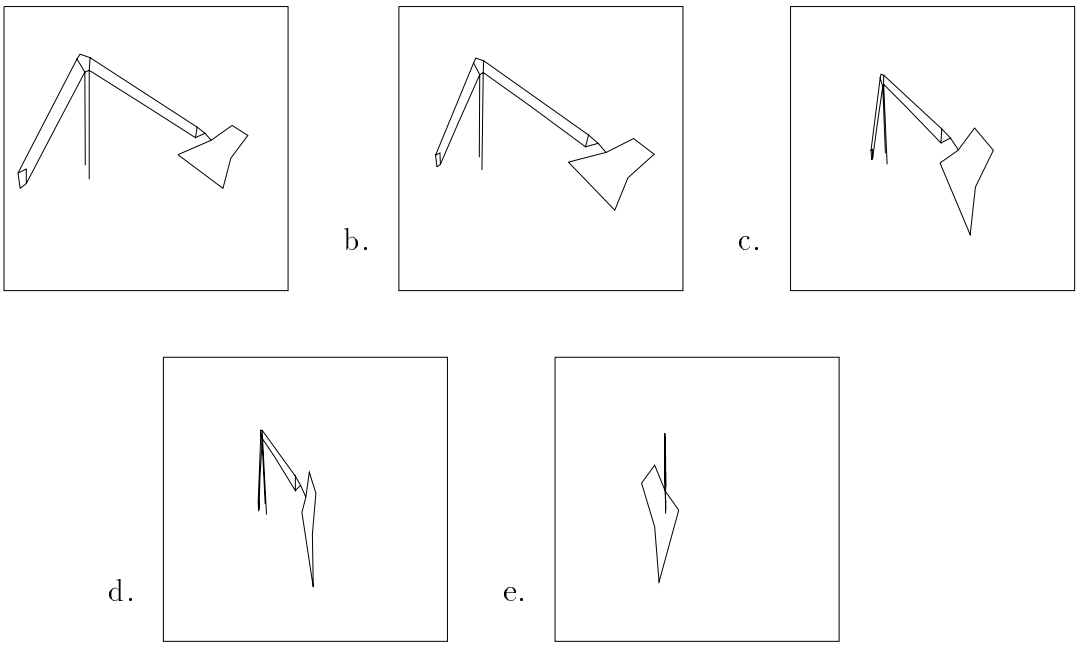

Figure 5.30: Several views of a synthesized lamp reconstructed from the 3D measurements and model constraints.

shown at the appropriate node where the value is the Mahalanobis distance δ calculated

using the folmula given is Section 5.7.1. For each level in the I.T. we show the three best scored nodes. The model constraints are fused during the parsing of the I.T. as described by the algorithm in Section 5.7.1. The distance constraints between model points k and j (denoted by $dist(k, j)$) are shown in the figure at the level at which they are fused. As can be seen the score of the correct matches are significally lower than the erroneous matches.

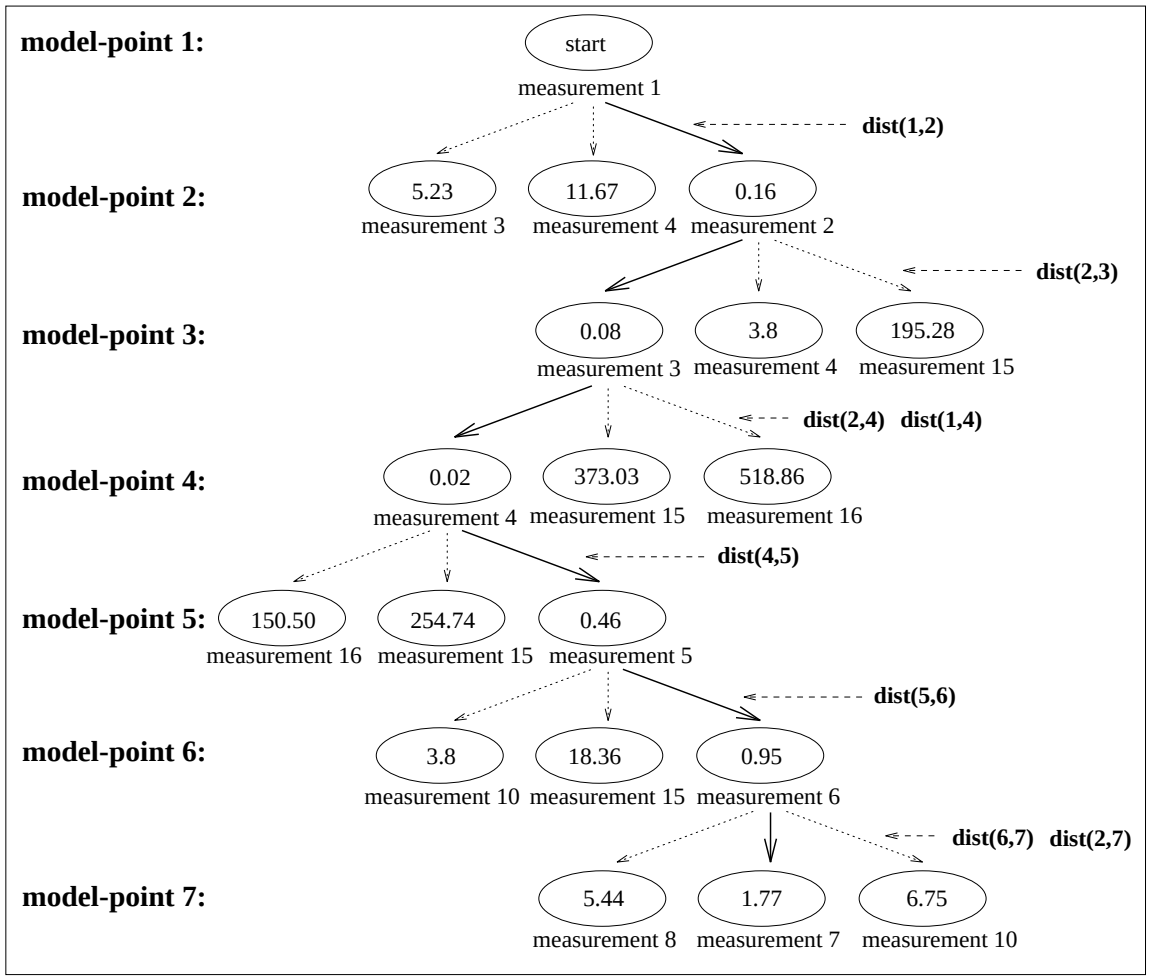

Figure 5.31: Results of the matching algorithm for the lamp model. A section of the pruned matching tree is displayed. Every level of the tree corresponds to one model point, and each node at a particulal level correspond to a possible match between the model point and a measured point. The score of each match is shown at the node where the value is the Mahalanobis distance of this match. For each level in the interpretation tree the three best scored nodes are shown. The distantce constraints (denoted by $dist(k, j)$) are shown at the level at which they are fused.

5.12 Constrained Objects: Conclusion

In this chapter we described three methods for estimating the pose of the components of a constrained model. The constraints are general and can be associated with any number of different parts of the model. The solution obtained by each of the three methods is optimal under the minimum variance criterion. Thus, these three methods give the same final solution. The validity of the framework was shown on real and simulated images.

The suggested method has several advantages over the previous methods:

1. In any pose estimation method, exploiting the information supplied by the measurement requires a definition of the functional dependence between the measurement and the estimated parameters. In the parametric methods (see Section 2.6), this dependence is not simple since it must include all the parameters on which the measurement depends. Additionally, the order of the nonlinearity of the dependence equations increases with the number of parameters. In our method, the functional dependence includes only the local parameters (i.e. only the parameters that define the transformation of the measured component) since the dependence of the measurement on other parameters is expressed through the constraints of the model. This local dependence is simply defined and is not as highly nonlinear as obtained by the parametric methods. Additionally, there is no need to reduce the constrained parameter space into a set of free parameters (as is performed in the parametric methods), thus the definition of the set of parameters to be estimated, is simple.

2. The information obtained on the position of a given component is propagated to all other components of the model through the constraints between them. Thus the estimated pose of a certain component takes into consideration all the existing measurements and all the defined constraints (this is not true in the divide and conquer methods).

3. The existing methods of pose estimation of constrained models, deal with articulated objects and with constraints that are due to prismatic or revolute joints between the model components. In our method we are not limited to any type

of constraints and can deal with all types of constraints including co-linearity, co-planarity, constant distance, constant angle, etc. Additionally, we deal with inequality constraints such as limited range of distances between points or limited range of angles.

4. The suggested solution uses K.F. tools and so includes the advantages associated with the K.F. such as explicitly dealing with the measurement uncertainty, simple updating of the solution given additional measurements, easy parallelization, and the possibility of using an efficient matching strategy.

Although we presented several strategies for finding an correspondence of the measurements, we concentrated, in this dissertation, on the problem of pose estimation of model components and did not give rigorous discussion nor implementation of the interpretation problem. Though both pose and correspondence problems are interlinked, we dealt with the problem of pose estimation separately in order to develop a rigorous and efficient technique for solving the problem.

The paradigm discussed in this work can be extended in several directions which were not discussed in this work:

- Complex primitives - We described the method for models composed of feature points. The method described can be extended to deal with models consisting of more complex primitives such as segments, lines, planes, etc. For every primitive type, one must define the set of parameters describing the position of the primitive and one must define the relation between the uncertainty of these parameters and the uncertainty of the measurements. Such an extension of our method is beneficial since measuring complex primitives contributes more information on the pose of the component, than measuring feature points.

- Flexible objects - The constraints we dealt with in this work were defined as "strict" constraints, i.e. "measurements" with zero uncertainty. Associating values greater than zero with the uncertainty of the constraint "measurements", will relax the strictness of the constraints and they will become "soft" constraints. This approach can be used in estimating the pose of elastic objects (such as rubber objects) or

objects with imprecise models. Imprecision in the model can arise from lack of knowledge on the model or when defining a general model representing a class of objects (such as a general model for a human skeleton).

- Hyerarchical models - In many cases it is convenient to describe the object model hierarchically, where primitives at higher levels represent global and prominent features and at lower levels represent features which are more local in nature. The hierarchy is characterized by the dependence of the primitives in the model on the primitives at higher levels. For example, the position of the eyes will be described in relation to the position of the head, and the position of the head in relation to the position of the body. Hierarchical description of objects has been widely dealt with in the literature [65, 23]. The hierarchical description of an object can reduce the complexity of estimating its pose and thus its detection; global features described at higher model levels are easier to synthesize and they supply a rough estimate of the pose of the object. This estimate will assist in reducing the search space of the primitives defined at lower model levels. Fusion of information from different hierarchical levels and definition of the search space of lower level primitives based on high level information, can be performed similar to the method described in this work for evaluating pose of articulated objects with a tree-like structure. The analogy is in that the relationships between primitives at different levels can be considered as mutual constraints.

We extended and implemented the ideas developed in this thesis in the directions described below.These implementations were not described in the thesis body, beeing out of context, but we briefly review them here.

- Application to keyframe animation and inverse kinematics - Keyframe animation deals with automatic generation of intermediate frames (denoted "inbetween" frames) based on a set of given key-frames. Classical approaches to generating inbetween-frames use linear interpolation of some control points in the key frames [18]. This approach fails when implemented on frames including articulated objects since the interpolation does not necessarily conserve the object constraints [75] (see example in Figure 5.32).

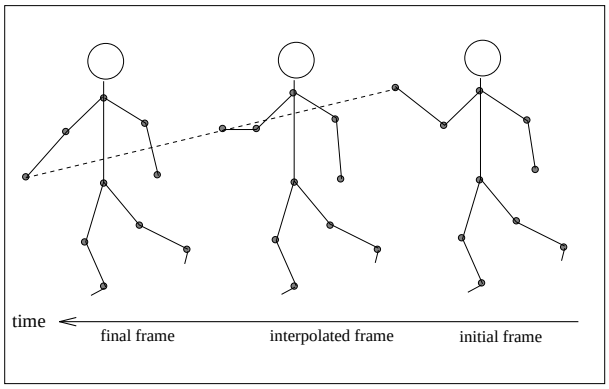

Figure 5.32: Linear interpolation of key frames does not conserve arm length.

The proposed method to solve this problem is to add uncertain "virtual measurements" to the control points in the key-frames. The interpolation will be performed on these virtual measurements rather than on the control points. The interpolated measurements will be used to position the inbetween-frame objects using the pose estimation method. Creating inbetween-frames, with this method, conserves the constraints of the semi-rigid objects. The smoothness of the object motion can be achieved by applying an additional dynamic Kalman filter process along the interpolated frames which acts as a damping force.

An example of an animation sequence is shown in Figure 5.33. The model to be animated consists of three control points (marked as gray circles) having 5 constraints beteen them: the two control points in the piston are constrained to lie on a horizontal line, the distance between them is constrained to be constant, the distance between the control point on the wheel and the middle control point is also constrained to be constant and the control point on the wheel is constrained to be at a constant distance from the center of the wheel. "Virtual measurements" are associated with each control point in the first (Figure 5.33a) and last (Figure 5.33h) frames of the sequence. The "virtual measurements" (marked as squares) are linearly interpolated between the first and last frame and the control points are positioned accordingly using the pose estimation method. Using "virtual measurements", the constraints of the system are conserved. For example the "virtual measurement" located on the wheel is an interpolation point (thus it moves linearly between frames) whereas the associated control point rotates around the

wheel, conserving the constraint of constant distance from the center of the wheel.

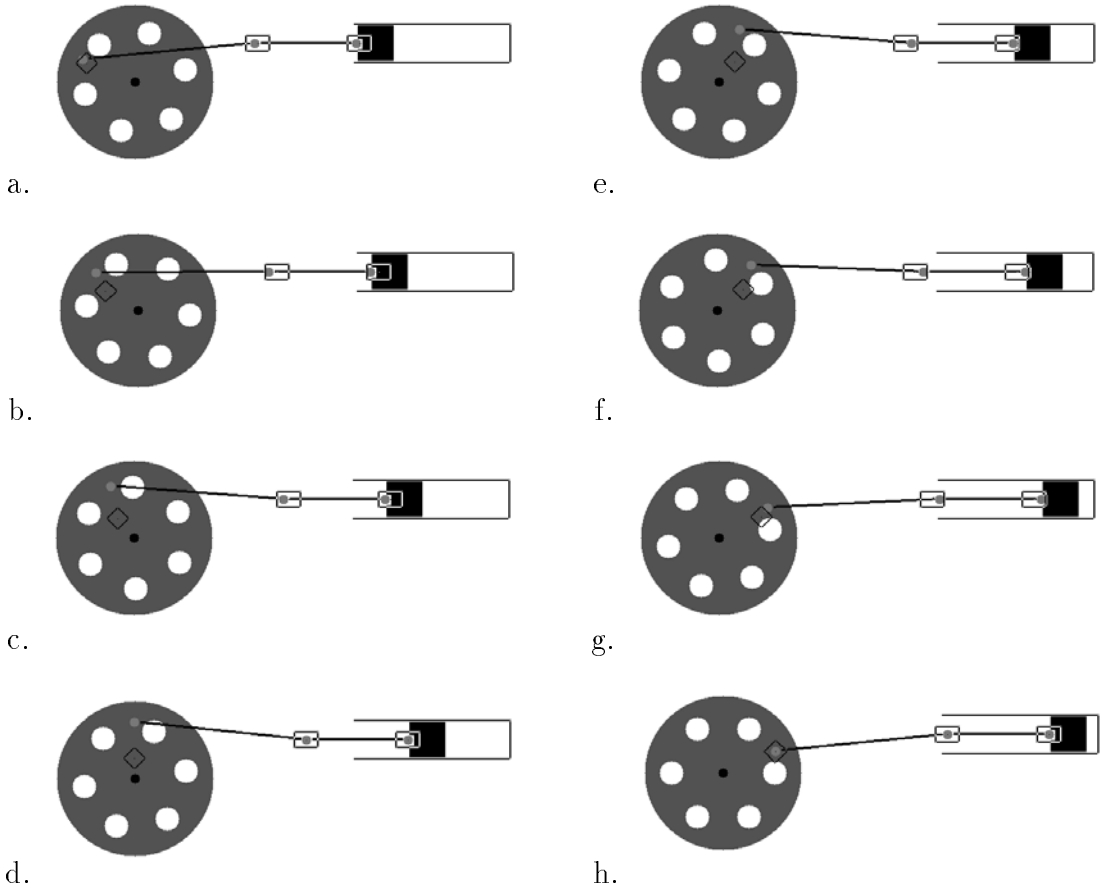

Figure 5.33: An example of an animation sequence using "virtual measurements" (see text). The control points are marked as gray circles, and the "virtual measurements" are marked as squares. Using "virtual measurements", the constraints of the system are conserved.

The animation problem is similar to the important problem of inverse kinematics in the area of robotics [78]; Given an initial and final position of the gripper of a robot arm, the aim is to find the motion of each robot component that will bring the gripper smoothly from initial to final position. This motion must satisfy the constraints defined by the mechanics of the robot arm. The approach we suggest to solve this problem follows the solution to the animation problem, as proposed above.

In this solution we set a small uncertainty to the virtual measurement associated with the gripper and large uncertainty to the virtual measurements associated with the other links. Therefore, the gripper will retain the smooth trajectory whereas the other links will relocate so that they follow a path of minimum change from one time frame to the next.

- Application to parametric modeling - Parametric design [47] is a paradigm in which the free parameters of a model are not determined directly by the designer; rather, a set of constraints (relations) is defined over parameters, and the system automatically computes the degrees of freedom and the free parameters. Parametric design is one of the most important development in mechanical computer aided design in the last few years and is commercially successful [61]. However, current parametric system require full and exact specification of the parameter constraints. Under- and over-constrained models are easy to produce, and manual correction of these is time consuming and error prone. Current parametric systems therefore cause over-specification and over-work.

  Using the proposed paradigm we suggest a design method we call *relaxed parametric design*. The method provides the designer with the capability of expressing *soft constraints*, constraints which do not have to be exactly met. Soft constraints are used whenever the designer wishes to express a general decision or guideline, avoiding over-specification. We implemented a parametric modeler based on the proposed paradigm. This work was published is [31].

Appendix A

Probability Theory: Background

For every random variable x with a continuous distribution, there exists a Probability Distribution Function (p.d.f.) $f(x)$ defined on $\Re$. $f(x)dx$ is the probability that the variable x lies in the interval $[x, x+dx]$, thus

$$\int_{-\infty}^{\infty} f(x)dx = 1$$

Similarly for a vector of random variables $\mathbf{x} = (x_1, x_2, \ldots)^t$ the p.d.f. is defined where $f(\mathbf{x})d\mathbf{x}$ is the probability that the vector $\mathbf{x}$ lies in the volume interval $[\mathbf{x} \times (\mathbf{x} + d\mathbf{x})] = [x_1, (x_1 + dx_1)] \times [x_2, (x_2 + dx_2)] \times \cdots$ and the following is satisfied:

$$\int_{-\infty}^{\infty} \ldots \int_{-\infty}^{\infty} f(\mathbf{x})dx_1, dx_2 \ldots = \int_{-\infty}^{\infty} f(\mathbf{x})d\mathbf{x} = 1$$

Expectation value:
The expectation value of a vector $\mathbf{x}$ with p.d.f. $f(\mathbf{x})$ is defined as:

$$E\{\mathbf{x}\} = \int_{-\infty}^{\infty} \mathbf{x} f(\mathbf{x})d\mathbf{x} = \bar{\mathbf{x}}$$

If $r(\mathbf{x})$ is a function of the vector $\mathbf{x}$ then

$$E\{r(\mathbf{x})\} = \int_{-\infty}^{\infty} r(\mathbf{x}) f(\mathbf{x})d\mathbf{x}$$

Variance and Covariance:
the covariance matrix of two vectors is defined as:

$$\mathrm{cov}\{\mathbf{x}, \mathbf{y}\} = E\{(\mathbf{x} - \bar{\mathbf{x}})(\mathbf{y} - \bar{\mathbf{y}})^t\} \stackrel{\mathrm{def}}{=} \Sigma_{\mathbf{xy}}$$

and specifically:

$$\mathrm{cov}\{\mathbf{x},\mathbf{x}\} = \mathrm{var}\{\mathbf{x}\} = E\{(\mathbf{x}-\bar{\mathbf{x}})(\mathbf{x}-\bar{\mathbf{x}})^t\} \stackrel{\mathrm{def}}{=} \Sigma_{\mathbf{x}}$$

Given an n-dimensional vector $\mathbf{x}$, the $n \times n$ matrix Σ_x is defined by its elements:

$$\begin{aligned} (\Sigma_{\mathbf{x}})_{ii} &= E\{(x_i-\bar{x}_i)^2\} \stackrel{\mathrm{def}}{=} \sigma_i^2 \\ (\Sigma_{\mathbf{x}})_{ij} &= E\{(x_i-\bar{x}_i)(x_j-\bar{x}_j)\} = \mathrm{cov}\{x_i,x_j\} \quad . \end{aligned} \tag{A.1}$$

σ_i^2 is the *variance* of the random variable x_i which is in fact the second central moment of the distribution of x_i and represents the "spread" of this distribution. $\mathrm{cov}\{x_i,x_j\}$ is the covariance of the random variables x_i and x_j and is equal to zero when these variables are independent. Thus, if the n variables of the vector $\mathbf{x}$ are mutually independent, the covariance matrix $\Sigma_{\mathbf{x}}$ will be diagonal. In any case $\Sigma_{\mathbf{x}}$ is a positive, semidefinite and symmetric matrix that can be diagonalized by a similarity transform.
The "spread" of the distribution of $\mathbf{x}$ can be represented by the trace of $\Sigma_{\mathbf{x}}$:

$$\mathrm{trace}(\Sigma_{\mathbf{x}}) = \sum_{i=1}^{n} \sigma_i^2 \quad .$$

This value is invariant under any similarity transformation.

Following the definition of the covariance we have the following equalities:

$$\begin{aligned} \mathrm{var}\{\mathbf{x}+\mathbf{y}\} &= \Sigma_{\mathbf{x}} + \Sigma_{\mathbf{y}} + 2\Sigma_{\mathbf{xy}} \\ \mathrm{var}\{A\mathbf{x}+\mathbf{b}\} &= A\Sigma_{\mathbf{x}}A^t \qquad \text{where } \mathbf{b} \text{ is a constant vector} \\ \mathrm{cov}\{\mathbf{x},A\mathbf{x}+\mathbf{b}\} &= \Sigma_{\mathbf{x}}A^t \end{aligned}$$

It can be shown [58] that if the random vector $\mathbf{y}$ is a non-linear function of the random vector $\mathbf{x}$, i.e.:

$$\mathbf{y} = r(\mathbf{x}) \quad ,$$

where $E\{\mathbf{x}\} = \bar{\mathbf{x}}$ and $\mathrm{var}\{\mathbf{x}\} = \Sigma_{\mathbf{x}}$ then the covariance $\Sigma_{\mathbf{y}}$ is approximated by:

$$\Sigma_{\mathbf{y}} \approx J\Sigma_{\mathbf{x}}J^t$$

where J is the Jacobian $\frac{\partial \mathbf{r}}{\partial \mathbf{x}}$ at the point $\mathbf{x} = \bar{\mathbf{x}}$.

The Normal Density

A common and important probability distribution is the *Normal* distribution often called the *Gaussian* distribution. The general multivariate normal density is defined as:

$$f(\mathbf{x}) = \frac{1}{(2\pi)^{\frac{d}{2}}\|\Sigma\|^{\frac{1}{2}}}\exp\{-\frac{1}{2}(\mathbf{x}-\mathbf{u})^t\Sigma^{-1}(\mathbf{x}-\mathbf{u})\} \tag{A.2}$$

where $\mathbf{x}$ is a d-component vector, $\mathbf{u}$ is a d-component constant vector, Σ is a $d \times d$ positive semidefinite matrix and $\|\Sigma\|$ is the determinant of Σ. This distribution is totally determined by $\mathbf{u}$ and Σ and is usually denoted by:

$$f(\mathbf{x}) = N(\mathbf{u}, \Sigma)$$

It can be shown that:

$$\begin{aligned} \mathbf{u} &= E\{\mathbf{x}\} \\ \Sigma &= \text{var}\{\mathbf{x}\} \end{aligned} \tag{A.3}$$

It is easily seen [19] that any linear combination of normally distributed random variables is itself normally distributed, i.e. if A is a $n \times d$ matrix and $\mathbf{y} = A\mathbf{x}$ is a n-component vector, where $\mathbf{x} = N(\mathbf{u}, \Sigma)$, then

$$f(\mathbf{y}) = N(A\mathbf{u}, A\Sigma A^t)$$

It follows from Eq. A.2 that the loci of points of constant density are hyper-ellipsoids for which the quadric form $(\mathbf{x}-\mathbf{u})^t\Sigma^{-1}(\mathbf{x}-\mathbf{u})$ is constant. The principal axis of these hyper-ellipsoids are given by the eigenvectors of Σ. The eigenvalues determine the length of these axes. The quantity $\mathbf{r}^2 = (\mathbf{x}-\mathbf{u})^t\Sigma^{-1}(\mathbf{x}-\mathbf{u})$ is sometimes called the squared *Mahalanobis distance* from $\mathbf{x}$ to $\mathbf{u}$. Thus the contours of constant density are hyper-ellipsoids of constant Mahalanobis distances to $\mathbf{u}$. It is common to graphically describe a bivariate normal density function by a contour of equal density having a predefined Mahanalobis distance value, such as $\mathbf{r}^2 = 1$.

Appendix B

Rotation Quaternion

A quaternion $\tilde{\mathbf{q}}$ is composed of two parts - the scalar part q_0 and the vector part $\mathbf{q}$:

$$\tilde{\mathbf{q}} = (q_0, \mathbf{q}) = (q_0, q_1 i + q_2 j + q_3 k) \quad .$$

If $\mathbf{u}$,$\mathbf{u}'$ are vectors in $\Re^3$ such that $\mathbf{u}' = R\mathbf{u}$, when R is a rotation matrix, then the corresponding expression in quaternion form is [36]:

$$\check{\mathbf{u}}' = \tilde{\mathbf{q}}\check{\mathbf{u}}\tilde{\mathbf{q}}^* \quad .$$

The quaternions $\tilde{\mathbf{u}}, \tilde{\mathbf{u}}'$ correspond to the vectors $\mathbf{u}$,$\mathbf{u}'$ respectively as follows:

$$\tilde{\mathbf{u}} = (0, \mathbf{u}) \quad ; \quad \tilde{\mathbf{u}}' = (0, \mathbf{u}')$$

and $\tilde{\mathbf{q}}^*$ is the conjugate of $\tilde{\mathbf{q}}$;

$$\tilde{\mathbf{q}}^* = (q_0, -\mathbf{q}) \quad .$$

$\tilde{\mathbf{q}}$ represents a rotation of the vector $\mathbf{u}$ by angle θ around a unit vector $\hat{\mathbf{n}}$ where:

$$q_0 = \cos(\frac{\theta}{2}) \quad ; \quad \mathbf{q} = \sin(\frac{\theta}{2})\hat{\mathbf{n}}$$

so that

$$\|\tilde{\mathbf{q}}\|^2 = \tilde{\mathbf{q}}\tilde{\mathbf{q}}^* = q_0^2 + \|\mathbf{q}\|^2 = 1 \quad .$$

Appendix C

Example of Batch Solution For an Articulated Object

Assume the articulated object as illustrated in Figure 5.17. This object has five point components $\{\mathbf{u}_i\}_{i=1}^{5}$ and four articulated constraints $\{\psi_j(\mathbf{T})\}_{j=1}^{4}$. Each constraint is of the form

$$\psi_j(\mathbf{t}_k, \mathbf{t}_l) = \|\mathbf{t}_l - \mathbf{t}_k\|^2 - d_{k,l}^2 = 0 \tag{C.1}$$

representing the constant Euclidean distance existing between the linked points $\mathbf{u}_k$ and $\mathbf{u}_l$. Additionally, assume a single measurement $(\hat{\mathbf{u}}'_k, \Lambda_k)$ for each model point $\mathbf{u}_k$. Linearization of Equation C.1 around $(\hat{\mathbf{u}}'_k, \hat{\mathbf{u}}'_l)$ yields:

$$\frac{1}{2}(d_{k,l}^2 + \mathbf{v}_{k,l}^2) = \mathbf{v}_{k,l}\mathbf{t}_l - \mathbf{v}_{k,l}\mathbf{t}_k$$

where

$$\mathbf{v}_{k,l} = \hat{\mathbf{u}}'_l - \hat{\mathbf{u}}'_k \quad .$$

Linearization of all the constraint equations and concatenating them into one vector equation will give:

$$\mathbf{z} = H\mathbf{T} \tag{C.2}$$

where

$$\mathbf{z} = \frac{1}{2}\begin{pmatrix} d_{1,2}^2 + \mathbf{v}_{1,2}^2 \\ d_{1,3}^2 + \mathbf{v}_{1,3}^2 \\ d_{3,4}^2 + \mathbf{v}_{3,4}^2 \\ d_{3,5}^2 + \mathbf{v}_{3,5}^2 \end{pmatrix} \quad ; \quad H = \begin{bmatrix} \mathbf{v}_{2,1} & -\mathbf{v}_{2,1} & 0 & 0 & 0 \\ \mathbf{v}_{3,1} & 0 & -\mathbf{v}_{3,1} & 0 & 0 \\ 0 & 0 & \mathbf{v}_{4,3} & -\mathbf{v}_{4,3} & 0 \\ 0 & 0 & \mathbf{v}_{5,3} & 0 & -\mathbf{v}_{5,3} \end{bmatrix} \quad ; \quad \mathbf{T} = \begin{pmatrix} \mathbf{t}_1 \\ \vdots \\ \mathbf{t}_5 \end{pmatrix} \quad .$$

Solving the illustrated example will be done by supplying the K.F. fuser with the following inputs:

- The *a priori* estimate for the evaluated transformation will be:

$$\left[\begin{pmatrix} \hat{\mathbf{u}}_1' \\ \vdots \\ \hat{\mathbf{u}}_5' \end{pmatrix}, \begin{pmatrix} \Lambda_1 & & 0 \\ & \ddots & \\ 0 & & \Lambda_5 \end{pmatrix} \right]$$

- The measurement input will be the vector $\mathbf{z}$ (of size 15) and its associated uncertainty - a 15×15 zero matrix.

- The measurement model input is Equation C.2 as formulated above.

Appendix D

Bibliography

Bibliography

[1] A. V. Aho, J. E. Hopcroft, and J. D. Ullman. *The Design and Analysis of Computer Algorithms*. Addison-Wesley, Reading, MA, 1974.

[2] B.D.O. Anderson and J.B. Moore. *Optimal Filtering*. Prentice-Hall, Inc., N.J., 1979.

[3] K.S. Arun, T.S. Huang, and S.D. Blostein. Least squares fitting of two 3D point sets. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 9(5):698–700, Sept. 1987.

[4] N. Ayache. *Artificial Vision for Mobile Robots Stereo Vision and Multisensory Perception*. MIT Press, 1991.

[5] N. Ayache and O.D. Faugeras. Hyper: A new approach for the recognition. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 8(1):44–54, Jan 1986.

[6] N. Ayache and O.D. Faugeras. Building, registrating, and fusing noisy visual maps. In *International Conf. on Computer Vision*, pages 73–82, 1987.

[7] N. Ayache and O.D. Faugeras. Maintaining representation of the environment of a mobile robot. *IEEE Trans. Robotics and Automation*, 5(6):804–819, Dec. 1989.

[8] H. Baker. Three-dimensional modeling. In *International Joint Conference on Artificial Intelligence*, pages 231–241, 1977.

[9] D.H. Ballard. Generalizing the hough transform to detect arbitrary shapes. *Pattern Recognition*, 13:111–122, 1981.

[10] R. Basri. *The Recognition of 3-D Solid Objects from 2-D Images*. PhD thesis, Weizmann Institute of Science, Oct. 1990.

[11] R. Basri and D. Weinshal. Distance metric between 3D models and 2D images. In *The 9th Israeli Symposium on Artificial Intelligence and Computer Vision*, pages 337–356, Dec. 1992.

[12] A. Beinglass and H.J. Wolfson. Articulated object recognition, or: How to generalize the generalized hough transform. In *Conference on Computer Vision and Pattern Recognition*, pages 461–466, 1991.

[13] T.O. Binford. Visual perception by computer. In *IEEE Conference on System and Control*, 1971.

[14] S.D. Blostein and T.S. Huang. Estimating 3-D motion from range data. In *Conf. on Artificial Intelligence Applications*, pages 246–250, 1984.

[15] R.M. Bolle and D.B. Cooper. On optimally combining pieces of information, with application to estimating 3D complex-object position from range data. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 8(5):619–638, Sept. 1986.

[16] R.C. Bolles and R.A. Chain. Recognizing and locating partially visible objects: The local feature focus method. *International Journal of Robotics Research*, 1(3):57–82, 1982.

[17] R.A. Brooks. Model-based 3-D interpretation of 2-D images. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 5:140–150, 1983.

[18] N. Burtnyk and M. Wein. Computer-generated key-frame animation. *SMPTE*, 80:149–153, 1971.

[19] M. H. DeGroot. *Probability and Statistics.* Addison-Wesley, Reading, MA, 1975.

[20] O.D. Faugeras, N. Ayache, and B. Faverjon. A geometric matcher for recognizing and positioning 3D rigid objects. In *Conference on Artificial Intelligence Applications*, pages 218–224, 1984.

[21] O.D. Faugeras and M. Hebert. A 3D recognition and positioning algorithm using geometrical matching between primitive surfaces. *International Joint Conference on Artificial Intelligence*, 2:996–1002, 1983.

[22] O.D. Faugeras and M. Hebert. The representation, recognition, and positioning of 3D shapes from range data. In A. Rosenfeld, editor, *Techniques for 3D Machine Perception*, pages 13–51. Elsevier Science, 1986.

[23] O.D. Faugeras and J. Ponce. Prism trees: A hierarchial representation for 3D objects. In *International Joint Conference on Artificial Intelligence*, pages 982–988, 1983.

[24] M.A. Fischler and R.C. Bolles. Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM*, 24(6):381–395, June 1981.

[25] S. Ganapathy. Decomposition of transformation matrices for robot vision. *Pattern Recognition Letters*, 2:401–412, 1984.

[26] W.E.L. Grimson. Recognition of object families using parameterized models. In *International Conf. on Computer Vision*, pages 93–101, 1987.

[27] W.E.L. Grimson. On the recognition of parametrized 2-D objects. *International Journal of Computer Vision*, 2:353–372, 1989.

[28] W.E.L. Grimson. On the verification of hypothesized matches in model-based recognition. In *European Conference on Computer Vision*, pages 489–498, 1990.

[29] W.E.L. Grimson and T. Lozano-Perez. Model-based recognition and localization from sparse range or tactile data. *International Journal of Robotics Research*, 3(3):3–35, 1984.

[30] R.M. Haralick, H. Joo, C.N. Lee, X. Zhuang, V.G. Vaidya, and M.B. Kim. Pose estimation from corresponding point data. *IEEE Trans. Systems, Man, and Cybernetics*, 19(6):1426–1445, Nov/Dec 1989.

[31] Y. Hel-Or, A. Rappoport, and M. Werman. Relaxed parametric design with probabilistic constraints. In *Solid Modelling-93*, 1993.

[32] Y. Hel-Or, A. Shmuel, and M. Werman. Active feature localization. In *Active Perception and Robot Vision*. Springer Verlag, 1991.

[33] Y. Hel-Or and M. Werman. Absolute orientation from uncertain data: A unified approach. In *Conference on Computer Vision and Pattern Recognition*, pages 77–82, 1992.

[34] D.M. Himmelblau. *Applied Nonlinear Programming*. McGraw-Hill, 1972.

[35] R.J. Holt and A.N. Netravali. Camera calibration problem: Some new results. *Computer Vision, Graphics and Image Processing*, 54(3):368–383, Aug 1991.

[36] B.K.P. Horn. *Robot Vision*. MIT Press, 1986.

[37] B.K.P Horn. Closed-form solution of absolute orientation using unit quaternion. *J. Opt. Soc. Am.*, 4(4):629–642, 1987.

[38] B.K.P. Horn, H.M. Hilden, and S. Negahdaripour. Closed-form solution of absolute orientation using orthonormal matrices. *J. Opt. Soc. Am.*, 5(7):1127–1135, July 1988.

[39] M.K. Hu. Visual pattern recognition by moment invariants. *IRE Transactions on Information Theory*, 8:169–187, Feb. 1962.

[40] P.J. Huber. *Robust Statistics*. John Wiley, New-York, 1981.

[41] D. Huttenlocher. *Three-Dimensional Recognition of Solid Objects from a Two-Dimensional Image*. PhD thesis, MIT AI Lab, April 1988.

[42] D.P. Huttenlocher and S. Ullman. Object recognition using alingment. In *International Conf. on Computer Vision*, pages 102–111, 1987.

[43] A.H. Jazwinski. *Stochastic Process and Filtering Theory*. Academic Press, 1970.

[44] G.A. Korn and T.M. Korn. *Mathematical Handbook for Scientists and Engineers*. McGraw-Hill, New-York, 1986.

[45] R. Kumar. *Model Dependent Inference of 3D Information from a Sequence of 2D Images*. PhD thesis, University of Massachusetts at Amherst, Feb. 1992.

[46] K. Levenberg. A method for the solution of certain non-linear problems in least-squares. *Quart. Appl. Math.*, 2:164–168, 1944.

[47] V.C. Lin, D.C. Gossard, and R.A. Light. Variational geometry in computer aided-design. *Computer Graphics*, 15(3):117–126, 1981.

[48] Z.C. Lin, T.S. Huang, S.D. Blostein, H. Lee, and E.A. Margerum. Motion estimation from 3-D point sets with and without correspondences. In *Conference on Computer Vision and Pattern Recognition*, pages 194–201, 1986.

[49] Y. Liu, T.S. Huang, and O.D. Faugeras. Determination of camera location from 2D to 3D line and point correspondences. In *Conference on Computer Vision and Pattern Recognition*, pages 82–88, 1988.

[50] D.G. Lowe. Three-dimensional object recognition from single two-dimensional images. *Artificial Intelligence*, 31:355–395, 1987.

[51] D.G. Lowe. Stabilized solution for 3-D model parameters. In *European Conference on Computer Vision*, pages 408–412, 1990.

[52] D.G. Lowe. Fitting parametrized 3-D models to images. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 13:441–450, May 1991.

[53] D.W. Marquardt. An algorithm for least-squares estimation of non-linear problems. *Journal Soc. Indust. Appl. Math.*, 11:431–441, 1963.

[54] L. Matthies and T. Kanade. The cycle of uncertainty and constraint in robot perception. *International Journal of Robotics Research*, 4, 1987.

[55] P.S. Maybeck. *Stochastic Models, Estimation, and Control*, volume 1. Academic Press, 1979.

[56] D.J. Meagher. Geometric modeling using octree encoding. *Comp. Graph. and Image Proc.*, 19(2):129–147, June 1981.

[57] I.J. Mulligan, A.K. Mackworth, and P.D. Lawrence. A model-based vision system for manipulator position sensing. In *Workshop on Interpretation of 3D Scenes, Austin, Texas*, pages 186–193, 1989.

[58] A. Papoulis:65. *Probability Random Variables and Stochastic Processes*. McGraw-Hill, 1965.

[59] A.P. Pentland. Perceptual organization and the representation of natural form. *Artificial Intelligence*, 28(3):290–331, 1986.

[60] J. Ponce and D.J. Kriegman. Geometric modeling using octree encoding. *International Journal of Computer Vision*, July 1989.

[61] S. Porter. Solid modeling: Winds of change. In *Computer Graphics World*, Feb. 1991.

[62] A. Requicha and H. Voelcker. Constructive solid geometry. Technical Report TR-26, University of Rochester, 1977.

[63] C.W. Richard. Identification of three dimensional objects using furier descriptors of the boundry curve. *IEEE Transactions on Systems, Man, and Cybernetics*, 4(4):371–378, 1974.

[64] B. Sabata and J.K. Aggarwal. Estimation of motion from a pair of range images: A review. *Computer Vision, Graphics and Image Processing*, 54(3):309–324, Nov 1991.

[65] H. Samet. The quadtree and related hierarchial data structures. *ACM Computing Surveys*, 16(2):187–260, 1984.

[66] T. Shakunaga. Pose estimation of jointed structures. In *Conference on Computer Vision and Pattern Recognition*, pages 566–572, 1991.

[67] S. Shekhar, O. Khatib, and M. Shimojo. Object localization with multiple sensors. *International Journal of Robotics Research*, 7(6):34–44, Dec. 1988.

[68] A. Shmuel and M. Werman. Active vision: 3D depth from an image sequence. In *International Conference on Pattern Recognition*, pages 48–54, 1990.

[69] D. Shoham and S. Ullman. Aligning a model to an image using minimal information. In *International Conf. on Computer Vision*, pages 259–263, 1988.

[70] A. Sluzek. Using moment invariants to recognize and locate partially occluded 2D objects. *Pattern Recognition Letters*, 7:253–257, 1988.

[71] G. Strang. *Introduction to applied mathematics.* Wellesley-Cambridge Press, 1986.

[72] T.M. Strat. Recovering the camera parameters from a transformation matrix. In *Proc. DARPA Image Understanding Workshop*, pages 264–271, 1984.

[73] J.H. Stuelpnagle. On the parameterization of the three-dimensional rotation group. *SIAM Review*, 6:422–430, 1964.

[74] I. Sutherland. Three-dimensional data input by tablet. *Proceedings of the IEEE*, 62:453–461, Apr 1974.

[75] D. Thalmann. Motion control: From keyframe to task-level animation. In *State-of-the-art in Computer Animation.* Springer-Verlag, 1989.

[76] R.Y. Tsai. A versatile camera calibration technique for high-accuracy 3D machine vision metrology using off-the-shelf tv cameras and lenses. *IEEE Journal of Robotics and Automation*, 3(4):323–344, Aug 1987.

[77] K.J. Udapa and I.S.N. Murthy. New concepts of three-dimensional shape analysis. *IEEE Transaction on Computation*, 26(10):1043–1049, Oct. 1977.

[78] M. Vukobratovic. *Introduction to Robotics.* Springer-Verlag, 1989.

[79] J. Weng, N. Ahuja, and T.S. Huang. Optimal motion and structure estimation. In *Conference on Computer Vision and Pattern Recognition*, pages 144–152, 1989.

[80] T.P. Wallace P.A. Wintz. An efficient 3D aircraft recognition algorithm using normalized furier description. *Computationa Graphics and Image Processing*, 13:96–126, 1980.

[81] J.S.C. Yuan. A general photogrammetric method for determining object position and orientation. *IEEE Transactions on Robotics and Automation*, 5(2):129–142, 1989.

[82] Z. Zhang. *Motion Analysis from a Sequence of Stereo Frames and its Applications.* PhD thesis, Universite de Paris-Sud, centre d'Orsay, Oct. 1990.

[83] Z. Zhang and O. Faugeras. *3D Dynamic Scene Analysis - A Stereo Based Approach.* Springer, Berlin, 1992.