

# Kinematic Sets for Real-time Robust Articulated Object Tracking

Andrew I. Comport, Éric Marchand, François Chaumette  
IRISA - INRIA Rennes  
Campus de Beaulieu, 35042 Rennes Cedex, France.  
E-Mail: `Firstname.Name@irisa.fr`

## Abstract

In this article a new approach is given for real-time visual tracking of a class of articulated non-rigid objects in 3D. The main contribution of this paper consists in symmetrically modeling the motion and velocity of an articulated object via a novel kinematic set approach. This is likened to a Lagrange-d'Alembert formulation in classical physics. The advantages of this new model over pre-existing methods include improved precision, robustness and efficiency, leading to real-time performance. Furthermore, a general class of mechanical joints can be considered and the method can track objects where previous approaches have failed due to a lack of visual information. In summary, a joint configuration is modeled by using Pfaffian velocity constraints. The configuration and location of a joint is then used to build a general Jacobian Matrix which relates individual rigid body velocities (twists) to an underlying minimal subspace. A closed loop control law is then derived in order to minimize a set of distance errors in the image and estimate the system parameters. The tracking is locally based upon efficient distance criterion. Experimental results show prismatic, rotational and helical type links and up to eight general parameters. A statistical M-estimation technique is applied to improve robustness. A monocular camera system was used as a real-time sensor to verify the theory.

Image and Vision Computing, IVC  
to appear in 2006

## 1 Introduction

In the past, a large amount of work has been invested in target tracking using images, with the tracking being based on measures such as color, texture, appearance, shape or combinations of these. Rigid 3D tracking methods have emerged which are both efficient and robust. On the other hand, tracking of non-rigid structures is a relatively new field with few publications focused purely on the aspect of visual perception of these types of objects. In this paper, a compact and symmetric based formulation is given for modeling any articulated object configuration and the proposed method is shown to be precise and perform in real-time in complex environments.

Non-rigid motion has been classed into three categories in [1] describing different levels of constraints on the movement of a body: articulated, elastic [21] and fluid [29]. Probabilistic methods such as the particle filter [15, 9] or Markov-chain based techniques [28] could be considered as under-constrained systems and research using these approaches seek to further constrain the system [13] so as to obtain more precise and less approximate estimates. Alternatively, rigid tracking methods can be considered as over-constrained systems and a bottom up approach would seek to relax these constraints so that a more general class of objects may be considered. Relatively few recent methods focus on the latter approach and this paper aims to exhibit advantages of model-based methods such as precision, efficiency, robustness to occlusion etc...

In this paper, the class of articulated non-rigid motion is considered. An “articulated” object is defined as a multi-body system composed of at least two rigid *components* and at most five independent degrees of freedom between any two components. With articulated motion, a non-rigid but constrained dependence exists between the components of an *object*. Previous methods have attempted to describe articulated motion either with or without an a-priori model of the object. In this study, a 3D model is used due to greater robustness and computational efficiencies. Knowing the object’s structure helps to predict hidden movement, which is particularly interesting in the case of articulated objects because there is an increased amount of self-occlusion. Knowing the model also allows an analytic relation for the system dynamics to be more precisely derived. Furthermore, it is easy to show that the formulation given in this paper provides a continuous and general basis for extending these techniques to elastic as well as fluid motion.

### 1.0.1 State of the Art

In general, the methods which have been proposed in the past for articulated object tracking rely on a good rigid tracking method. In computer vision, the geometric primitives considered for tracking have been numerous, however, amongst them distance based features have shown to be efficient and robust [17, 11, 4]. Another important issue is the 2D-3D registration problem. *Purely geometric* (eg, [10]), or *numerical and iterative* [7] approaches may be considered. *Linear approaches* use a least-squares method to estimate the pose and are considered to be more suitable for initialization procedures. *Full-scale non-linear optimization techniques* (e.g., [17, 19, 11, 4]) consist of minimizing the error between the observation and the forward-projection of the model. In this case, minimization is handled using numerical iterative algorithms such as Newton-Raphson or Levenberg-Marquardt. The main advantage of these approaches are their accuracy. The main drawback is that they may be subject to local minima and, worse, divergence. This approach is better suited to maintaining an estimate of state parameters for real-time robust tracking of an object because local minima are avoided when the errors remain small. The drawback of this approach being that it needs to be initialized.

Within this context, it is possible to envisage different ways to model motion of an articulated object. A first class of methods for visual tracking of articulated objects uses kinematic chains [8] (see Figure 1 (a)). A good example appears in work by Lowe [18]. He demonstrates a method using partial derivatives of image features with respect to object pose and articulation joint parameters which vary with time. In his paper, the kinematic chain of articulations is represented as tree structure of internal rotation and translation

parameters and the model points are stored in the leaves of this tree. The position and partial derivatives of each point in camera-centered coordinates is determined by the transformations along the path back to the root.

Recently, more complex features have been used for non-rigid object tracking in [23]. They make use of deformable super-quadric models combined with a kinematic chain approach. However, real-time performance is traded-off for more complex models. Furthermore, this method requires multiple viewpoints in order to minimize the system of equations. As Lowe points out, the tendencies in computer graphics have been toward local approximations via polyhedral models. Ruff and Horaud [26] give another kinematic-chain style method for the estimation of articulated motion with an un-calibrated stereo rig. They introduce the notion of projective kinematics which allows rigid and articulated motions to be represented within the transformation group of projective space. The authors link the inherent projective motions to the Lie-group structure of the displacement group. The minimization is determined in projective space and is therefore invariant to camera calibration parameters.

Another recent approach has been proposed by Drummond and Cippola [11] which treats articulated objects as groups of rigid components with constraints between them, expressed directly in camera coordinates (see Figure 1 (b)). In this formulation, the full pose of each rigid component is initially computed independently requiring the estimation of a non-minimal number of parameters. In a second step, Lagrange multipliers are then used to constrain these parameters according to simple link definitions. This method uses Lie Algebra to project the measurement vector (distances) onto the subspace defined by the Euclidean transformation group (kinematic screw). They implement M-estimation to improve robustness.

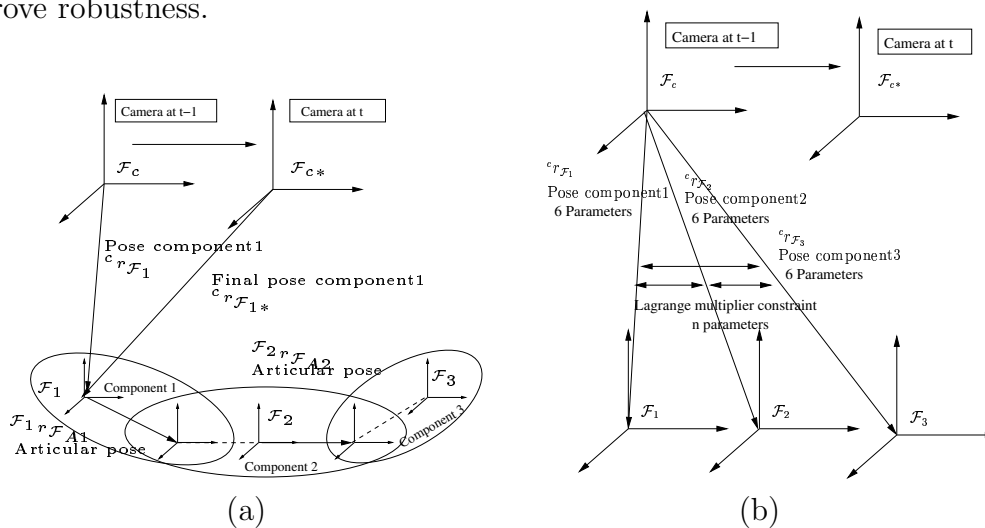


Figure 1: Two different models of articulated motion, (a) Kinematic chain method: The pose of an articulated object is determined via a kinematic chain of rigid bodies extending to sub components. Each circle represents a rigid component. The root component is rigidly linked to the camera via the pose  ${}^c\mathbf{r}_{\mathcal{F}_1}$  and each subsequent component is linked by the parameters contained within a pose such as  ${}^{\mathcal{F}_1}\mathbf{r}_{\mathcal{F}_2}$  from component 1 to 2. (b) Lagrange Multiplier method: The pose between the camera and each part of the object  ${}^c\mathbf{r}_{\mathcal{F}_1}$ ,  ${}^c\mathbf{r}_{\mathcal{F}_2}$ ,  ${}^c\mathbf{r}_{\mathcal{F}_3}$  are calculated directly in a first step. Constraints between the components are then enforced in a second minimization step via Lagrange multipliers. In this case, a kinematic chain or tree is also used to represent the articulated structure.

## 1.0.2 Contribution

In this paper, a new method is proposed for real-time 3D tracking of articulated objects. The registration problem is handled using full-scale non-linear minimization. This minimization can be seen as the dual problem of visual servoing whereby minimizing the parameters corresponds to controlling an arm-to-eye system so as to observe the arm at a given position in the image (note that an object is not necessarily fixed to the ground). This duality is known as Virtual Visual Servoing [4, 20].

The advantages of the rigid body tracking method, presented in this paper, include:

- Highly reactive spatial tracking without prediction or filtering lag.
- Real-time performance due to a 1D search and correspondence strategy involving one-to-many feature correspondences.
- Accurate and precise due to an iteratively re-weighted least squares minimization procedure known as Virtual Visual Servoing.
- Robust to occlusion, miss-tracking, background clutter, large changes in illumination and other general sources of error. Robustness is due to a known CAD model and statistical M-estimation.
- 3D model allows to describe the complete pose space and articulated pose space and can handle Hidden Surface Removal (HSR).

Apart from these advantages, the major contribution of this paper is the definition of a new *articulated* 3D model representation which can be traced back to the Lagrange-Alembert theory in classical physics. This theory states that instead of applying constraints which apply forces to a system, the coordinate basis is changed to be orthogonal to the constraint so that the only forces in the system are the real physical forces [25]. However, in terms of modeling and tracking an articulated object, it is only necessary to derive the equations in terms of 3D location and orientation velocities as opposed to the more complicated equations of potential and kinetic energy involved in this theory.

When compared with existing Kinematic Chain approaches, it can be seen that there is no need to sum partial derivatives along a kinematic chain back to the root in a visual system as the camera has a *direct* visual link with the movement of each component of an articulated object and not only the root component. As will be shown, the joint reference frame also plays an important role in modeling articulated objects. The joint reference frame can be thought of as the point of intersection between two rigid components which together make up an articulated object. The method presented in this paper also defines a general mechanical joint formulation for simple definition of articulations.

The formalism proposed here is called a “kinematic set” approach as opposed to a chain or tree approach. In this case, a novel subset approach is used whereby the minimization is carried out on decoupled subsets of parameters by defining subspace projectors from joint definitions. This allows the error seen in the image to be partially decoupled from the velocities of the object by determining the independent sets of velocities present.

In motion estimation problems, there exists an inherent duality between the motion of the sensor and motion of the scene. More precisely, the motion between a single camera sensor and a rigid object can be attributed to the movement of either the camera or the object. Similarly, in the case of multiple objects, the movement can equally be considered as movement of multiple camera sensors. Furthermore, with articulated motion, unlike

the case of rigid motion, the subsets of movement which may be attributed to either the object or camera are not unique. In this article, the focus will be placed on a monocular camera observing a constrained multi-body object, however, these methods may equally be applied to the case of multiple cameras [6, 24].

The principal advantages of the kinematic set approach are:

- Any type of mechanical link can be considered (i.e. helical, translational, rotational, ball joint etc...).
- allows tracking of under-constrained or rank deficient components (e.g:  $< 3$  points).
- eliminates the propagation of errors between free parameters (Important for noisy sensor measurements and approximation).
- is efficient in terms of computation as well as representation space.
- models closely the real behavior of the system. (i.e. multiple objects or cameras).

In the remainder of this paper, Section 2 presents the principle of the approach. In Section 3, articulated object motion and velocities are defined. In Section 4, a closed loop control law is derived for tracking articulated objects. In Section 5, several experimental results are given for different objects.

## 2 Overview and Motivations

The objective of the tracking approach is to maintain an estimate of the set of minimal parameters  $\mathbf{q} \in \mathbb{R}^m$ . This is a minimal vector or set of  $m$  parameters describing the configuration of an articulated object in  $SE(3)^k$  where  $k$  is the number of rigid components making up and articulated object. This space is defined so as to incorporate all different types of articulated objects including the case where both components are completely independent.  $SE(3)$  or the Special Euclidean Group is the configuration space of a rigid body and is also known as a Lie Group. An element of  $SE(3)$  is referred to as a pose or location and belongs to a 6-dimensional differential manifold.

The modeling of articulated object motion is first based on rigid body differential geometry. Consider the 3D Euclidean space with the related vector space being  $\mathbf{r} \in \mathbb{R}^6$ . This is defined as a vector  $\mathbf{r} = (t_x, t_y, t_z, \theta_x, \theta_y, \theta_z)$ , which is composed of translational  $t$  and rotational components  $\theta$ . The tangent space to  $SE(3)$  is the Lie Algebra  $se(3)$  of velocities. An element of  $se(3)$  is known as a twist. This is defined as a vector  $\mathbf{v} = (v_x, v_y, v_z, \omega_x, \omega_y, \omega_z)$ , which is composed of linear  $v$  and angular  $\omega$  velocity components. The Lie Group is defined so that Euclidean transformations preserve distances and orientations of object features with respect to one another. Both these vector quantities are related by Rodriguez's exponential map [3]  $\mathbf{r} = exp(\mathbf{v})$ .

Alternatively to rigid bodies, non-rigid bodies belong to a  $m$ -dimensional configuration space  $\mathbf{q} \in SE(3)^k$  in which the extra degrees of freedom describe internal parameters allowing intra-object movement. It is a  $m$ -dimensional differential manifold. The tangent space of velocities is denoted  $se(3)^k$ . In this article, we will be mainly interested in the tangent space. An element of this generalized configuration space,  $\dot{\mathbf{q}} \in \mathbb{R}^m$ , is defined as  $\dot{\mathbf{q}} = (\dot{q}_1, \dots, \dot{q}_m)$ . For this generalized configuration space, the Lie Group defines Euclidean transformations which preserve different *subsets* of distances and orientations of object features. The subsets of velocity parameters within this manifold are also related to their Special Euclidean counterpart by Rodriguez's exponential map.

In order to maintain an estimate of  $\mathbf{q}$ , the underlying idea is to minimize a non-linear system of equations so that the projected contour of the object model in the image is aligned with the actual position of the contours in the image. To perform the alignment, an error  $\Delta$  is defined in the image.

$$\Delta = \left( \mathbf{s}(\mathbf{q}) - \mathbf{s}^* \right) \quad (1)$$

where  $\mathbf{s}(\mathbf{q})$  is a vector of feature parameters projected from a 3D model onto the image and  $\mathbf{s}^*$  is a vector of corresponding features found by a tracking procedure in the image (desired features). In this paper efficient distance features are used (see Sections 3.2 and 5.1).

At the base of the algorithm is an a-priori CAD model for any type of *multi-body object*. 3D parametric equations describe the structure of a rigid component with respect to an object reference frame  $\mathcal{F}_o$ . Articulated objects are represented as rigid components (or feature sets) with defined joints between them. Each joint has an associated reference frame  $\mathcal{F}_i, \forall i = 1, \dots, l$ , where  $l$  is the number of joints. The representation of a joint or link is a core part of this paper and it will be derived in detail in Section 3.

In this paper, a monocular *camera sensor* is used where the center of projection is referenced by the camera reference frame  $\mathcal{F}_c$ . The formation of the image is a standard perspective projection model. The error in equation (1) can be written as:

$$\Delta = \left[ pr(\mathbf{q}, {}^o\mathbf{S}) - \mathbf{s}^* \right], \quad (2)$$

where  ${}^o\mathbf{S}$  are the 3D parameters of any type of visual feature associated with an object which are expressed in the object frame of reference  $o$ .  $pr(\mathbf{q}, {}^o\mathbf{S})$  is the camera projection model according to the generalized pose parameters  $\mathbf{q}$ . The features of each component are projected onto the image using their associated camera poses  ${}^{\mathcal{F}_c}\mathbf{r}_{\mathcal{F}_i}(\mathbf{q})$  where each component's camera pose is composed of a 6 dimensional subset of general parameters  $\mathbf{q}$ .

For the very first image at  $t_0$ , the parameters of the object are initially needed and they can be computed using different globally convergent methods. These include manually making the correspondence of four point features with the model [7], a manual manipulation of the object projected on the image (using the mouse) or an automatic procedure in the case of textured objects [16]. The initial estimate is then refined using the Virtual Visual Servoing [4, 20] non-linear iterative minimization approach. Following an approximate initialization procedure, the *closed-loop* tracking procedure proposed in this paper takes over. Since a closed-loop method is employed, any errors in the initialization procedure are eliminated. The advantages of using a non-linear minimization procedure include accuracy, robustness, and real-time efficiency. Although non-linear iterative methods are not globally convergent, they are ideal for tracking since any inter-frame movement of the object is relatively small. Even if large movements are observed within the image, the estimation process still converges since a large cone of convergence exists around the previous solution. A proof of convergence is given in Section 4.

In order to render the minimization of these errors more robust, they are minimized using a robust approach based on M-estimation techniques.

$$\Delta_{\mathcal{R},i} = \rho\left(s_i(\mathbf{q}) - s_i^*\right), \quad (3)$$

where  $\rho(u)$  is a robust function [14] that grows sub-quadratically and is monotonically nondecreasing with increasing  $|u|$ . See Appendix 1 for an overview of this technique.

### 3 Modeling : Articulated and Non-rigid Motion

The aim of this section is to describe and define articulated motion. In particular, the minimal parameter, generalized velocity vector is defined and derived. This is a non-trivial issue and the derivation of the model continues through the following four Sub-Sections.

Section 3.1 first considers an overview of the mapping required to pass from multiple individual rigid velocity vectors to a generalized velocity vector. Section 3.2 determines an Interaction matrix which relates distance to contour type image features to the movement of a rigid object. In Section 3.3, the definition of a joint is made including velocity constraint matrices as well as joint locations. This provides the basis for creating subspace projection operators for each joint. Following this, the joint subspaces are combined in a combinatorial fashion so as to determine all the kinematic sets of an object. Projection operators are then defined for each kinematic set and are finally combined (Section 3.4) in order to build a Jacobian mapping between the individual velocities of each component and the generalized velocities. This mapping is referred to as the Articulation matrix.

#### 3.1 The Generalized Twist

Our approach aims at reducing multiple *rigid-body* six dimensional velocity twists  $\mathbf{v}_1, \dots, \mathbf{v}_k$  describing the motion of  $k$  rigid components to a minimal set of parameters  $\dot{\mathbf{q}}$  describing the configuration of an articulated object.

First, consider the interaction matrix which defines the rigid-body mapping between a vector of visual feature velocities,  $\mathbf{s}_d$ , and the velocity twist,  $\mathbf{v}$ , containing linear and angular velocities of the pose as given in Section 2. Any kind of geometrical feature can be considered for measuring motion in the image as soon as it is possible to compute its corresponding interaction matrix  $\mathbf{L}_s$ . In [12], a general framework to compute  $\mathbf{L}_s$  is proposed. Indeed, it is possible to compute the pose from a large set of image information (points, lines, circles, quadratics, distances, etc...) within the same framework. In this paper distance features are used to represent many to one correspondences between local point features and the contours of a projected CAD model (see Section 5.1). The combination of different features is achieved by adding features to vector  $\mathbf{s}$  and by “stacking” each feature’s corresponding interaction matrix into a large interaction matrix of size  $df_i \times 6$  where  $f_i$  corresponds to the number of features in component  $i$  and  $d$  is the dimension of the feature (eg: distance  $d = 1$ , point  $d = 2$ , line  $d = 2$ , ellipse  $d = 5$ , etc...):

$$\begin{bmatrix} \dot{\mathbf{s}}_1 & \dots & \dot{\mathbf{s}}_{df_i} \end{bmatrix}^\top = \begin{bmatrix} \mathbf{L}_1 & \dots & \mathbf{L}_{df_i} \end{bmatrix}^\top \mathbf{v}_i \quad (4)$$

The redundancy yields an accurate final estimation and allows simultaneous estimation using all available information. Furthermore, if the number or the nature of visual features is modified over time, the interaction matrix  $\mathbf{L}_s$  and the vector error  $\mathbf{s}$  is easily modified.

The configuration of a *multi-body* articulated object is completely defined by a differential mapping from  $(se(3))^k$  to  $\mathbb{R}^m$ , where  $m$  is the minimum number of parameters and ( $m < 6k$ ). The central idea is to relate the movement  $\dot{\mathbf{s}}$  of the set of sensor features to

the movement of the generalized object parameters. This is defined as:

$$\begin{bmatrix} \dot{s}_1 \\ \vdots \\ \dot{s}_{f_1} \\ \vdots \\ \dot{s}_n \end{bmatrix} = \begin{bmatrix} \begin{bmatrix} L_{1,1}^1 & \cdots & L_{1,6}^1 \\ \vdots & \ddots & \\ L_{f_1,1}^1 & & L_{f_1,6}^1 \end{bmatrix} & \mathbf{0} & \begin{bmatrix} A_{1,1} & \cdots & A_{1,m} \\ \vdots & \ddots & \\ A_{6 \times k,1} & & A_{6 \times k,m} \end{bmatrix} \\ \mathbf{0} & \begin{bmatrix} L_{1,1}^k & \cdots & L_{1,6}^k \\ \vdots & \ddots & \\ L_{f_k,1}^k & & L_{f_k,6}^k \end{bmatrix} & \begin{bmatrix} \dot{q}_1 \\ \vdots \\ \dot{q}_m \end{bmatrix} \end{bmatrix}, \quad (5)$$

where  $n$  is the number of image features,  $f_i$  is the number of features in rigid component  $i$ ,  $k$  is the number of components, and  $m$  is the minimal number of parameters.

This is summarized in matrix form as:

$$\dot{\mathbf{s}} = \mathbf{L}_D \mathbf{A} \dot{\mathbf{q}} \quad (6)$$

where

- $\mathbf{L}_D \in \mathbb{R}^{n \times 6k}$  is composed of multiple interaction matrices [12],  $\mathbf{L}_{s_i}$ , along the diagonal. The different rigid bodies or components have a corresponding interaction matrix (4) with  $\dot{\mathbf{s}}_i = \mathbf{L}_{s_i} \mathbf{v}_i$  for all  $i = 1 \dots k$ . Each block's column width is of dimension 6 corresponding to the six parameters of each  $\mathbf{v}_i$ . The row vectors each correspond to one of  $n$  visual feature parameters.
- $\mathbf{A} \in \mathbb{R}^{6k \times m}$  is an Articulation matrix describing the differential relation between component's velocities  $\mathbf{v}_i$  and the minimal parameter subspace  $\dot{\mathbf{q}}$ . This matrix is composed of different subspaces which will be derived in the remainder of this Section.
- $\mathbf{L}_D \mathbf{A}$  is the Jacobian between the visual features and the configuration of entire object.

This Articulation matrix is the central issue and is defined as the mapping  $\mathbb{R}^{6k} \rightarrow \mathbb{R}^m$ :

$$\mathbf{v} = \mathbf{A} \dot{\mathbf{q}} \quad (7)$$

where  $\mathbf{v} \in (se(3))^k$  is a vector of 'stacked' 6-dimensional twists each corresponding to the full velocity twist in  $se(3)$  of each component.

This series of definitions implies that relative environmental modifications of the geometrical kind are the only ones allowed to vary. Each component can be represented individually as a rigid body and there exists a minimal subspace related to the entire object's geometrical variations.

The subsets of parameters which make up the object parameters are illustrated by a Venn diagram in Figure 3.1. In order that the generalized sets can be obtained, it is necessary to find the basis vectors representing the minimal subspace. As will be shown in the following sections, this depends on the configuration of the object.

## 3.2 Interaction matrices

The derivation of the interaction matrix that links the variation of the distance between a fixed point and a moving straight line to the rigid velocity twist is now given. In Figure 3

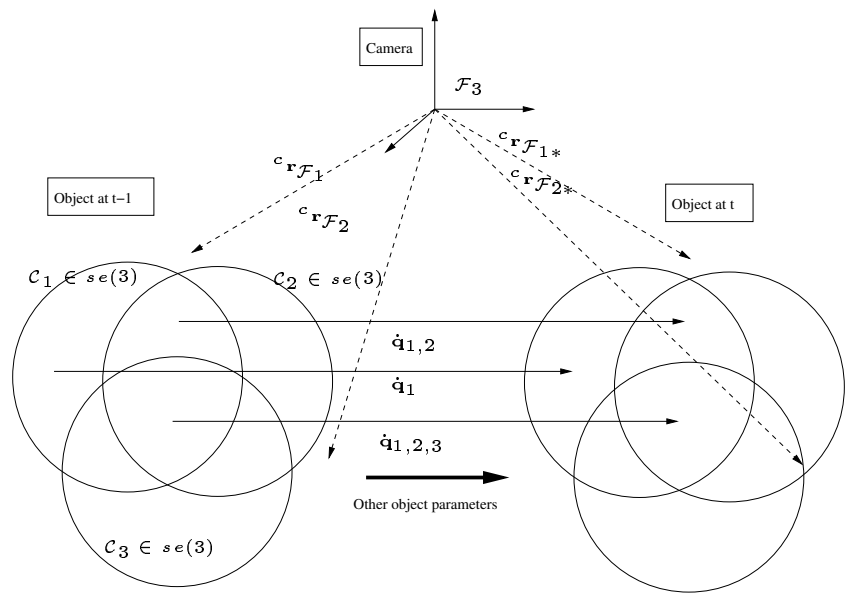


Figure 2: Kinematic set method: The generalized articulated object parameters correspond to the different subsets in the Venn diagram. Each circle corresponds to a set of 6 rigid velocity parameters  $\mathbf{v}$  which are related to the pose via the exponential map. The poses  $\mathbf{r}$  which are a non-minimal representation can be extracted from the minimal set  $\mathbf{q}$  and used to project an object CAD model onto the image. The minimal parameter set is minimized in a common reference frame and the kinematic sets model partially decouples the system. Decoupling occurs at the intersection of parameter sets.

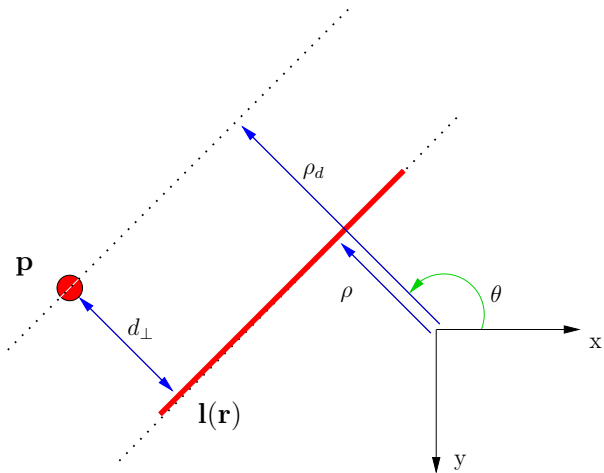


Figure 3: Distance of a point to a line. The line  $\mathbf{l}$  is projected onto the image using the pose  $\mathbf{r}$  and a point  $\mathbf{p}$  is found by a 1D search to the normal. No information is held in the direction perpendicular to the line and this correspondence forms the basis of a distance-to-line image feature.

$\mathbf{p}$  is the tracked point feature position and  $\mathbf{l}(\mathbf{r})$  is the line feature position. The position of the line is given by its polar coordinates representation,

$$x \cos \theta + y \sin \theta = \rho, \forall (x, y) \in \mathbf{l}(\mathbf{r}). \quad (8)$$

The distance between a point  $\mathbf{p}$  and  $\mathbf{l}(\mathbf{r})$  is characterized fully by the distance  $d_{\perp}$  perpendicular to the line. The point found to the normal of the line does not necessarily correspond to the departure point on the line for the edge detection procedure. In other words the distance parallel to the segment does not hold any useful information unless a correspondence exists between a point on the line and  $\mathbf{p}$  (which is in general not the case). Thus the distance feature from a line is given by:

$$d_l = d_{\perp}(\mathbf{p}, \mathbf{l}(\mathbf{r})) = \rho(\mathbf{l}(\mathbf{r})) - \rho_d, \quad (9)$$

where  $\rho_d = x_d \cos \theta + y_d \sin \theta$ , with  $x_d$  and  $y_d$  being the coordinates of the tracked point. Thus, the variation of the distance with time can be related to the variation of the line parameters by:

$$\dot{d}_l = \dot{\rho} - \dot{\rho}_d = \dot{\rho} + \alpha \dot{\theta}, \quad (10)$$

where  $\alpha = x_d \sin \theta - y_d \cos \theta$ .

Using the relation (10), it is clear that, the interaction matrix for a distance feature  $d_l$  can be deduced from the interaction matrix for a line  $\mathbf{L}_{d_l} = \mathbf{L}_{\rho} + \alpha \mathbf{L}_{\theta}$ , where the interaction matrix related to a straight line is given by (see [12] for its complete derivation):

$$\begin{aligned} \mathbf{L}_{\theta} &= \begin{bmatrix} \lambda_{\theta} \cos \theta & \lambda_{\theta} \sin \theta & -\lambda_{\theta} \rho & \rho \cos \theta & -\rho \sin \theta & -1 \end{bmatrix} \\ \mathbf{L}_{\rho} &= \begin{bmatrix} \lambda_{\rho} \cos \theta & \lambda_{\rho} \sin \theta & -\lambda_{\rho} \rho & (1 + \rho^2) \sin \theta & -(1 + \rho^2) \cos \theta & 0 \end{bmatrix} \end{aligned} \quad (11)$$

where  $\lambda_{\theta} = (A_2 \sin \theta - B_2 \cos \theta)/D_2$ ,  $\lambda_{\rho} = (A_2 \rho \cos \theta + B_2 \rho \sin \theta + C_2)/D_2$ , and  $A_2 X + B_2 Y + C_2 Z + D_2 = 0$  is the equation of a 3D plane which the line belongs to.

Evaluating the interaction matrix for a distance-to-line using the interaction matrix for a line-to-line, the following result is obtained:

$$\mathbf{L}_{d_l} = \begin{bmatrix} \lambda_{d_l} \cos \theta & \lambda_{d_l} \sin \theta & -\lambda_{d_l} \rho & (1 + \rho^2) \sin \theta - \alpha \rho \cos \theta & -(1 + \rho^2) \cos \theta - \alpha \rho \sin \theta & -\alpha \end{bmatrix}, \quad (12)$$

where  $\lambda_{d_l} = \lambda_{\rho} + \alpha \lambda_{\theta}$ .

Note that the case of a distance between a point and the projection of a cylinder or a portion of an ellipse is similar and the reader can refer to [4].

### 3.3 Joint Configuration

The general form of the articulation matrix, that links the rigid velocity twists of different components, can be derived by first considering the definition of a joint. Any type of joint or link can be fully defined by a velocity constraint matrix and a pose of the joint with respect to a fixed reference frame.

A joint matrix for joint  $j$  is denoted by  $\mathbf{J}_j(\mathbf{S}_j^{\perp}, \mathbf{r}_j)$ , where:

$\mathbf{S}_j^{\perp}$  is a basis matrix, which remains constant over time, defining the mechanical constraint of the joint  $j$ . This is defined and derived further in Section 3.3.1 and a detailed example is given in Section 3.3.2.

$\mathbf{r}_j(t)$  is a 6-parameter pose vector representing the location and orientation of the joint in 3D which varies over time. This is used to further derive the different parts of the articulation matrix in Section 3.3.3 and the first example is continued in Section 3.3.4.

### 3.3.1 Velocity Constraints

In this Section the mechanical configuration of a joint is defined by a static constraint matrix. More particularly, a velocity constraint matrix is used to define two subspaces of a single rigid mechanical link. One belonging to the rigid part of the link and the other to the non-rigid part.

In mechanics, a holonomic constraint can be used to restricts the motion of a rigid body to a smooth manifold. In this case, the constraint is represented locally as an algebraic constraint on the configuration space [22].

In mechanics, a holonomic constraint can be used to restricts the motion of a rigid body to a smooth manifold. In this case, the constraint is represented locally as an algebraic constraint on the configuration space [22]. This requires algebraically defining the space of joint positions by defining allowable positions. However, since the estimation of the pose parameters is usually carried out using the tangent space of velocities, it is easier to consider a constraint in the form of a velocity constraint.

Velocity constraints are known as Pfaffian constraints [25] and constrain velocities as:

$$\mathbf{J}^+(\mathbf{r})\mathbf{v} = 0, \quad \mathbf{J}(\mathbf{r}) \in \mathbb{R}^{c \times 6}, \quad (13)$$

where the constraint matrix  $\mathbf{J}$  represents  $c$  velocity constraints and  $\mathbf{J}^+ = (\mathbf{J}^\top \mathbf{J})^{-1} \mathbf{J}^\top$  is the pseudo-inverse of  $\mathbf{J}$ . According to the definition of a Pfaffian constraint, if it is holonomic, it is therefore integrable and it can be used to determine an algebraic constraint for the pose of a component if needed.

In order to simplify the determination of the velocity constraint matrix, in a first instance let us ignore the fact that a joint has a specific position in 3D space and concentrate on the static mechanical configuration of the joint. A link's mechanical configuration is fully defined by constraining a single component's twist vector. The constraint matrix is written as a standard basis:

$$\mathbf{S}_j^\perp = \begin{bmatrix} \frac{\partial S_1}{\partial v_1} & \cdots & \frac{\partial S_c}{\partial v_1} \\ \vdots & \ddots & \\ \frac{\partial S_1}{\partial v_6} & & \frac{\partial S_c}{\partial v_6} \end{bmatrix}, \quad (14)$$

where this is a holonomic constraint matrix and is defined such that each column vector defines one degree of freedom at the corresponding joint  $j$ . As mentioned previously, the number of non-zero columns of  $\mathbf{S}_j^\perp$  is referred to as the *class*  $c$  of the link. The rows of a column define the type of the link by defining which combination of translations and rotations are permitted as well as their proportions.

Let  $\mathbf{S}_j^\perp$  be the Image matrix subspace so that its corresponding Kernel subspace is given by (with abuse of notation):

$$\mathbf{S}_j = Ker((\mathbf{S}_j^\perp)^\top). \quad (15)$$

In terms of the physical joint, the set of velocities that a first component can undertake which leaves a second component invariant is defined by  $S^\perp \subset se(3)$ . This is the orthogonal complement of the sub-space  $S \subset se(3)$  which constitutes the velocities which are in common between two components. Since a component, that is linked to another, is composed of these two subspaces, it is possible to project component twists onto these subspaces with these standard bases for the kernel and the image.

It should be noted that due to the assignment of sets to matrices, the column vectors of the matrices should be considered as unordered and dependent upon numerical calculation methods. This unknown order is avoided by defining projection operators. In the following, the matrix  $\mathbf{S}_j$  and its orthogonal complement  $\mathbf{S}_j^\perp$ , defined in equations (14) and (15), are used to project the kinematic twist (velocities) onto two orthogonal subspaces. In Section 3.4, the more complicated case of more than one link is considered.

A pair of subspace projection matrices are given as:

$$\mathbf{P}_j = \mathbf{S}_j \mathbf{S}_j^+ \quad \text{and} \quad \mathbf{P}_j^\perp = \mathbf{S}_j^\perp \mathbf{S}_j^{\perp+} = \mathbf{I}_6 - \mathbf{P}_j, \quad (16)$$

where  $\mathbf{S}^+$  the pseudo-inverse of  $\mathbf{S}$ . This ensures that the resulting projected velocities are defined according to the common basis  $\mathbf{v}$  (see Section 2). Most importantly, this allows traditional twist transformations to be applied to these quantities.

### 3.3.2 Examples

In the experiments reported in the results section (Section 5), two different types of class 1 links are considered in Figure 3.3.2. Their corresponding constraint matrices are given in Table 1 and the application of these constraints to the velocity vector are given in Table 2.

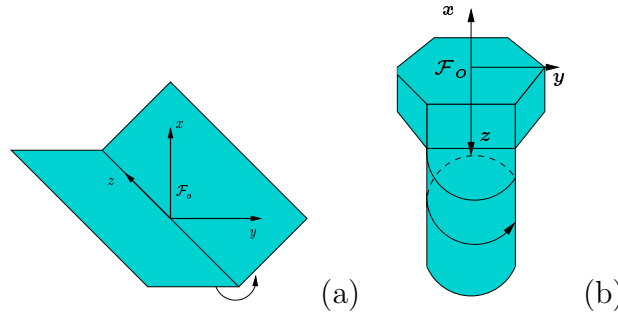


Figure 4: (a) A class one rotational hinge link between two rectangles, (b) A class one helical link between a screw and a plate (not shown).

For the example of a helical link, defined in equation (3.3.2), the mechanical configuration projector becomes:

$$\mathbf{P}_j = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & \frac{1}{a^2+1} & 0 & 0 & -\frac{a}{a^2+1} \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & -\frac{a}{a^2+1} & 0 & 0 & \frac{a^2}{a^2+1} \end{bmatrix}, \quad \mathbf{P}_j^\perp = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & \frac{a^2}{a^2+1} & 0 & 0 & \frac{a}{a^2+1} \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & \frac{a}{a^2+1} & 0 & 0 & \frac{1}{a^2+1} \end{bmatrix}. \quad (17)$$

$$\mathbf{S}_j^\perp = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \\ 0 \\ 0 \end{bmatrix}, \quad \mathbf{S}_j = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}, \quad \mathbf{S}_j^\perp = \begin{bmatrix} 0 \\ 0 \\ a \\ 0 \\ 0 \\ 1 \end{bmatrix}, \quad \mathbf{S}_j = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & -\frac{1}{a} \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix},$$

(a) (b)

Table 1: The velocity constraint matrix: (a) A **rotational link** around the x axis, (b) A **helical link** around and along the z axis which has a slightly different form. In this case the value of 'a' relates the translation along the z axis to a one rotation around the z axis. Note that  $\mathbf{S}_j$  is determined automatically by determining the Nullspace of  $\mathbf{S}_j^\perp$ .

$$\begin{aligned} (\mathbf{S}_j^\perp)^+ \mathbf{v}_j &= \omega_x, & (\mathbf{S}_j^\perp)^+ \mathbf{v}_j &= av_z + \omega_z, \\ \mathbf{S}_j^+ \mathbf{v}_j &= (v_x, v_y, v_z, \omega_y, \omega_z). & \mathbf{S}_j^+ \mathbf{v}_j &= (v_x, v_y, \omega_x, \omega_y, -\frac{1}{a}v_z + \omega_z). \end{aligned}$$

Table 2: Application of the constraints in Table (1) to the velocity vector  $\mathbf{v}$ .

It can be seen here that the columns are not linearly independent, however, the columns respect orthogonality with respect to the chosen frame parametrization.

### 3.3.3 Link Location and the Adjoint Map

Here the pose of the joint is introduced so as to fully define the joint constraint matrix  $\mathbf{J}$ . Firstly, a link position and orientation is fully defined by a pose vector:

$$\mathbf{r}_l = {}^{\mathcal{F}_c} \mathbf{r}_{\mathcal{F}_j} = (t_x, t_y, t_z, \theta_x, \theta_y, \theta_z), \quad (18)$$

where  $\mathcal{F}_c$  indicates the camera frame and  $\mathcal{F}_j$  represents the joint frame.

Since the previously defined mechanical constraint projector  $\mathbf{P}_j$  is only valid in the link frame of reference, it is necessary to determine the kinematic twist velocities as seen at the joint reference frame. This is achieved by a generic kinematic twist transformation matrix which is used to obtain the velocity of the link frame with respect to the camera. The Lie algebra provides the adjoint matrix  $\mathbf{V}(\mathbf{r})$  for transformation of vector quantities. This is a kinematic twist transformation from frame  $a$  to frame  $b$  given as:

$${}^{\mathcal{F}_a} \mathbf{V}_{\mathcal{F}_b} = \begin{bmatrix} {}^{\mathcal{F}_a} \mathbf{R}_{\mathcal{F}_b} & [{}^{\mathcal{F}_a} \mathbf{t}_{\mathcal{F}_b}]_{\times} {}^{\mathcal{F}_a} \mathbf{R}_{\mathcal{F}_b} \\ 0_3 & {}^{\mathcal{F}_a} \mathbf{R}_{\mathcal{F}_b} \end{bmatrix}, \quad (19)$$

where  ${}^{\mathcal{F}_a} \mathbf{R}_{\mathcal{F}_b}$  is a rotation matrix between frames and  ${}^{\mathcal{F}_a} \mathbf{t}_{\mathcal{F}_b}$  a translation vector between frames which are both obtained from the pose  ${}^{\mathcal{F}_a} \mathbf{r}_{\mathcal{F}_b}$ .

Thus, it is now possible to apply the joint projector defined in (16) in the link's spatial reference frame. In the implementation of this method, an arbitrary choice must be made between defining the position of the joint with respect to either the object frame of reference or the camera frame of reference. Furthermore, minimization must

$$\begin{aligned}
{}^c\mathbf{V}_o &= \begin{bmatrix} 1 & 0 & 0 & 0 & -1 & 0 \\ 0 & 1 & 0 & 1 & 0 & -1 \\ 0 & 0 & 1 & 0 & 1 & 0 \\ & \mathbf{0}_3 & & & \mathbf{I}_3 & \end{bmatrix} & \mathbf{J}_j &= \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & -\frac{1}{a} \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}, & \mathbf{J}_j^\perp &= \begin{bmatrix} 0 \\ -1 \\ a \\ 0 \\ 0 \\ 1 \end{bmatrix} \\
& \text{(a)} & & \text{(b)}
\end{aligned}$$

Table 3: Determining the joint subspace projection matrix for a particular example. (a) The adjoint matrix  ${}^c\mathbf{V}_o$ , (b) the joint projection matrices for a helical joint are determined by using equations (17), (a) and (20).

be performed in a common reference frame and therefore the camera frame is a natural choice. The interested reader can refer to previous works [5] where the object reference frame was used.

These choices lead to a generic subspace projection operator which defines the two subspaces of a joint as:

$$\begin{aligned}
\mathbf{J}_j &= \text{Im}({}^{\mathcal{F}_c}\mathbf{V}_{\mathcal{F}_j} \mathbf{P}_j {}^{\mathcal{F}_j}\mathbf{V}_{\mathcal{F}_c}), \\
\mathbf{J}_j^\perp &= \text{Ker}(\mathbf{J}_j) = \text{Im}({}^{\mathcal{F}_c}\mathbf{V}_{\mathcal{F}_j} \mathbf{P}_j^\perp {}^{\mathcal{F}_j}\mathbf{V}_{\mathcal{F}_c}),
\end{aligned} \tag{20}$$

where  $\text{Im}$  represents the Image operator which reduces the column space to its mutually independent basis form. The first transformation  ${}^{\mathcal{F}_c}\mathbf{V}_{\mathcal{F}_j}$  maps the rigid velocities of a component to the link frame  $l$  where the mechanical constraint is applied. The second transformation  ${}^{\mathcal{F}_j}\mathbf{V}_{\mathcal{F}_c}$  then re-maps the subspace velocities back to the common camera reference frame. The projector between features on a rigid object being  $\mathbf{J}_j = \mathbf{I}_6$  and for two completely independent objects  $\mathbf{J}_j = \mathbf{0}_6$ .

### 3.3.4 Examples

For simplicity, let the object be 1 meter along the z-axis and 1 meter along the x-axis of frame  $\mathcal{F}_c$  and is in front of the camera. By definition, the object reference frame is chosen to be the same as the joint reference frame for this example. The resulting velocity transformation matrix and the joint subspace projectors are given in Table (3). Since the transformation is orthogonal, the rank of the matrix's column space before and after transformation stays the same. Taking the Kernel and the Image of the resulting matrices as in (20), the dimension of the column space becomes equal to its rank.

## 3.4 Generalized Object Subspaces: Kinematic Sets

The aim of this section is to define the form of the articulation matrix  $\mathbf{A}$  which itself will define how all joint subspaces are tied together. In particular, the two previously defined subspaces of each joint ( $\mathbf{J}_j, \mathbf{J}_j^\perp$ ) are combined with the subspaces of other joints giving a certain number of subspaces for the entire object. This will in-turn define a basis for the minimum number of parameters which define the objects configuration.

An intuitive interpretation of the different subspaces can be obtained when referring to Figure 3.1. With one joint and two components, there are three distinct sub-

spaces, the subspace in common between the components plus the independent subspace of each component. With two joints and subsequently one virtual joint, there are possibly seven distinct subspaces. This is determined by looking at all the intersections and non-intersections between each subspace. It can be deduced that an articulated object that contains  $k$  components has possibly  $2^k - 1$  distinct subspaces. Likewise each individual component has  $2^{(k-1)}$  possible subspaces which are in common with the other components.

## Virtual Links

To maintain the geometric symmetry of the problem, it is necessary to have a description of the links between all rigid bodies of an object. If no link has been defined in the model, a 'virtual link' exists such that:

$$\mathbf{J}_{ab} = \bigcap_{\forall l \in \mathcal{P}} \mathbf{J}_l, \quad \text{and} \quad \mathbf{J}_{ab}^\perp = \bigcap_{\forall l \in \mathcal{P}} \mathbf{J}_l^\perp = \text{Ker}(\mathbf{J}_{ab}), \quad (21)$$

where  $\mathcal{P}$  is a path of joints connecting component  $a$  to  $b$  and where the intersection between two subspaces, represented by the basis matrices  $\mathbf{U}$  and  $\mathbf{V}$ , is defined as:

$$\mathbf{U} \cap \mathbf{V} = \text{Ker} \left( \mathbf{U}^\perp, \mathbf{V}^\perp \right). \quad (22)$$

The result of this intersection being an unordered basis for the intersecting subspace.

In order to determine all the subspaces of an object and eventually determine the form of the articulation matrix  $\mathbf{A}$ , it is necessary to use the intersection operator to create projectors for all combinations of joint subspaces. First, consider the basic cases of two and three components.

Let the set  $\mathcal{O}$  be the set of all combinations of joint subspaces for the entire object, where  $\dim(\mathcal{O}) = 2^k - 1$ . For two components and subsequently one joint,  $\mathcal{O}$ , excluding the empty-set  $\emptyset$ , is given by:

$$\begin{aligned} \mathcal{O} &= \{(\mathbf{v}_1 \cap \mathbf{v}_2), (\mathbf{v}_1 \notin \mathbf{v}_2), (\mathbf{v}_2 \notin \mathbf{v}_1)\} \setminus \emptyset, \\ &\equiv \{\mathbf{J}_{12}, \mathbf{J}_{12}^\perp, \mathbf{J}_{12}^\perp\} \setminus \emptyset, \end{aligned} \quad (23)$$

where in the first line each  $\mathbf{v}_i \in se(3)$  and the intersection between two sets is related to, but not equal to, the intersection operator in equation (22). In the second line each element refers to a joint with  $l = 12$  being the joint between component 1 and component 2. Note also that the subspace  $\mathbf{J}_{12}^\perp$  is the same for both  $(\mathbf{v}_1 \notin \mathbf{v}_2)$  and  $(\mathbf{v}_2 \notin \mathbf{v}_1)$ .

For a multi-body system composed of three components, there always exist three joints if virtual joints are included. The reason that virtual joints need to be determined is to provide a set of symmetrical equations with a general form. Furthermore, this allows the formulation to take into account redundant and parallel architectures. In this case,  $\mathcal{O}$  is given by:

$$\begin{aligned} \mathcal{O} &= \{(\mathbf{v}_1 \cap \mathbf{v}_2 \cap \mathbf{v}_3), (\mathbf{v}_1 \cap \mathbf{v}_2 \notin \mathbf{v}_3), (\mathbf{v}_1 \cap \mathbf{v}_3 \notin \mathbf{v}_2), (\mathbf{v}_2 \cap \mathbf{v}_3 \notin \mathbf{v}_1), \\ &\quad (\mathbf{v}_1 \notin \mathbf{v}_2 \notin \mathbf{v}_3), (\mathbf{v}_2 \notin \mathbf{v}_1 \notin \mathbf{v}_3), (\mathbf{v}_3 \notin \mathbf{v}_1 \notin \mathbf{v}_2)\} \setminus \emptyset, \\ &\equiv \{(\mathbf{J}_{12} \cap \mathbf{J}_{23} \cap \mathbf{J}_{31}), (\mathbf{J}_{12} \cap (\mathbf{J}_{23}^\perp \cap \mathbf{J}_{31}^\perp)), (\mathbf{J}_{23} \cap (\mathbf{J}_{12}^\perp \cap \mathbf{J}_{31}^\perp)), (\mathbf{J}_{31} \cap (\mathbf{J}_{12}^\perp \cap \mathbf{J}_{23}^\perp)), \\ &\quad (\mathbf{J}_1^\perp \cap \mathbf{J}_2^\perp \cap \mathbf{J}_3^\perp), (\mathbf{J}_1^\perp \cap \mathbf{J}_2^\perp \cap \mathbf{J}_3^\perp), (\mathbf{J}_1^\perp \cap \mathbf{J}_2^\perp \cap \mathbf{J}_3^\perp)\} \setminus \emptyset, \end{aligned} \quad (24)$$

where the order of the intersection operator and non-intersection are commutative.

Thus using boolean algebra, the set  $\mathcal{O}$  can be written generally for  $k$  components as:

$$\mathcal{O} = \wp \{ \mathbf{v}_1, \dots, \mathbf{v}_k \} \setminus \emptyset, \quad (25)$$

where  $\wp$  represents the intersection of all the combinations of subsets, selected from  $\mathbf{v}_1$  to  $\mathbf{v}_k$ , which are not an element of the remaining sets and not including the empty-set.

These sets can now be used to define projectors for each subspace of a component corresponding to a component's 'row block' of the articulation matrix (7). Furthermore, these sets must be ordered so that each column for each component is aligned with the corresponding subsets of the other components.

Using this definition, all subspace projectors are defined by the following simple equation:

$$\forall (s_o \subset \mathcal{O}), \mathbf{Q}_{s_o} = \left( \bigcap_{\forall \{ \mathbf{v}_a, \mathbf{v}_b \} \in s_o} \mathbf{J}_{ab} \right) \cap \left( \bigcap_{\forall \{ \mathbf{v}_a, \mathbf{v}_b \} \notin s_o} \mathbf{J}_{ab}^\perp \right), \quad (26)$$

where  $s_o$  is a subset of the object set  $\mathcal{O}$  and  $\cap$  is the subset intersection operator, defined in equation (22). Note that this equation creates the link between the components and the joints(including virtual joints).

### Articulation matrix

The articulation matrix fully defines a generalized coordinate basis for representing the minimal parameter vector  $\dot{\mathbf{q}}$ . In general terms, the Articulation Matrix corresponds to:

$$\mathbf{A} = \left( \frac{\partial \mathbf{r}_1}{\partial \mathbf{q}}, \dots, \frac{\partial \mathbf{r}_k}{\partial \mathbf{q}} \right)^\top. \quad (27)$$

The different subspaces determined in equation (26) are to be mapped to each component's subspace  $\mathcal{C}_i$ . In the case of a three component system, as in equation (24), component 1's subset would be given as:

$$\begin{aligned} \mathcal{C}_1 &= \{ (\mathbf{v}_1 \cap \mathbf{v}_2 \cap \mathbf{v}_3), (\mathbf{v}_1 \cap \mathbf{v}_2 \notin \mathbf{v}_3), (\mathbf{v}_1 \cap \mathbf{v}_3 \notin \mathbf{v}_2), (\mathbf{v}_1 \notin \mathbf{v}_2 \notin \mathbf{v}_3) \} \\ &\equiv \{ (\mathbf{J}_{12} \cap \mathbf{J}_{23} \cap \mathbf{J}_{31}), (\mathbf{J}_{12} \cap (\mathbf{J}_{23}^\perp \cap \mathbf{J}_{31}^\perp)), (\mathbf{J}_{31} \cap (\mathbf{J}_{12}^\perp \cap \mathbf{J}_{23}^\perp)), (\mathbf{J}_1^\perp \cap \mathbf{J}_2^\perp \cap \mathbf{J}_3^\perp) \}. \end{aligned} \quad (28)$$

The Articulation matrix is defined according to equation (7) and taking into account the joint subspaces given by equation (20). The Articulation Matrix can thus be written as:

$$\forall (i = 1..k) \text{ and } \forall (s_o \in \mathcal{O}) \quad \mathbf{A}_{i,s_o} = \delta_{\mathbf{v}_i \in s_o} \mathbf{Q}_{s_o}, \quad \text{where, } \delta = \begin{cases} 1 & \text{if } \mathbf{v}_i \in s_o, \\ 0 & \text{otherwise,} \end{cases} \quad (29)$$

where  $(i, s_o)$  is in (row, column) format and indexes the elements of the Articulation matrix.

The columns of  $\mathbf{A}$  are grouped in  $2^{k-1}$  sub-matrices of dimension  $p$ , each corresponding to a particular subset of object parameters. The rows are all in sub-groups of dimension 6 each corresponds to the six degrees freedom of each component.

The decoupling effect within the articulation matrix is directly related to the zeros represented by the  $\delta$  operator in equation (29). In particular, there are more zeros

introduced into the articulation matrix than in the case of a kinematic chain. In the case of a kinematic chain, zeros are only found in the upper diagonal of the articulation matrix, whereas, in the kinematic set case there are zeros both above and below the diagonal. This has an effect on the computational efficiency as well as the propagation of measurement error.

### 3.4.1 Example

Consider an object with three components ( $k = 3$ ). The first component has a x-rotational link to the second component 1m along the camera's z axis. The second component has a y-rotational link to the third 1m along the first joint's x axis (refer to Figure 5 (a)).

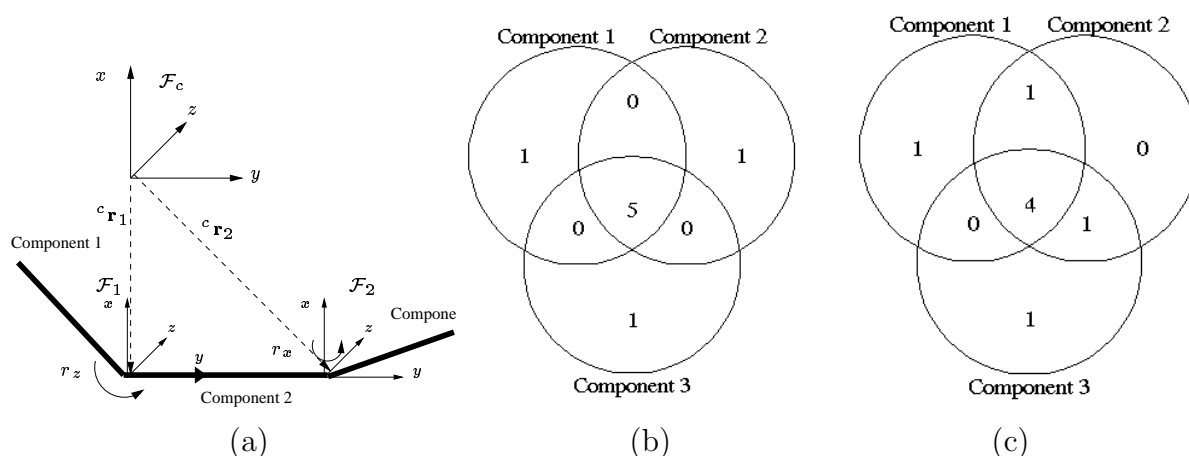


Figure 5: Example of a three body system, (a) Physical configuration of a camera and the three body system. (b,c) Dimensions of kinematic sets for a two component object with two rotational links. (b) Link locations are not orthogonal (c) Link locations are orthogonal

After computation of the different subspaces, the dimension of the objects subspaces are given in Figure 5(b) and (c). It is interesting to note that different subsets are decoupled automatically when the link reference frames are in an orthogonal position. It can also be noted that  $(\mathbf{J}_{31} \cap (\mathbf{J}_{12}^{\perp} \cap \mathbf{J}_{23}^{\perp}))$  is always equal to zero, unless there is a physical link defined between component one and component three.

A Maple program was written to prove the subset orthogonality properties of different mechanical and spatial configurations. It automatically creates the articulation matrix from joint definitions  $(\mathbf{S}_j, \mathbf{r}_j)$ . The Venn diagram, shown in Figure 5(b) and (c), displays the dimensions of the different subspaces which have been created from the program. It can be downloaded from: <http://www.irisa.fr/lagadic/code-eng.html>

For an object with three components and two joints, and using the orthogonal subspace projectors given in equation (26),  $\mathbf{A}$  is given by:

$$\mathbf{A} = \begin{pmatrix} \frac{\partial \mathbf{r}_1}{\partial \mathbf{q}_{123}} & \frac{\partial \mathbf{r}_1}{\partial \mathbf{q}_{12}} & \mathbf{0} & \frac{\partial \mathbf{r}_1}{\partial \mathbf{q}_{31}} & \frac{\partial \mathbf{r}_1}{\partial \mathbf{q}_1} & \mathbf{0} & \mathbf{0} \\ \frac{\partial \mathbf{r}_1}{\partial \mathbf{q}_{123}} & \frac{\partial \mathbf{r}_1}{\partial \mathbf{q}_{12}} & \frac{\partial \mathbf{r}_1}{\partial \mathbf{q}_{23}} & \mathbf{0} & \mathbf{0} & \frac{\partial \mathbf{r}_1}{\partial \mathbf{q}_2} & \mathbf{0} \\ \frac{\partial \mathbf{r}_1}{\partial \mathbf{q}_{123}} & \mathbf{0} & \frac{\partial \mathbf{r}_1}{\partial \mathbf{q}_{23}} & \frac{\partial \mathbf{r}_1}{\partial \mathbf{q}_{31}} & \mathbf{0} & \mathbf{0} & \frac{\partial \mathbf{r}_1}{\partial \mathbf{q}_3} \end{pmatrix} = \begin{pmatrix} \mathbf{Q}_{123} & \mathbf{Q}_{12} & \mathbf{0} & \mathbf{Q}_{31} & \mathbf{Q}_1^{\perp} & \mathbf{0} & \mathbf{0} \\ \mathbf{Q}_{123} & \mathbf{Q}_{12} & \mathbf{Q}_{23} & \mathbf{0} & \mathbf{0} & \mathbf{Q}_2^{\perp} & \mathbf{0} \\ \mathbf{Q}_{123} & \mathbf{0} & \mathbf{Q}_{23} & \mathbf{Q}_{31} & \mathbf{0} & \mathbf{0} & \mathbf{Q}_3^{\perp} \end{pmatrix}, \quad (30)$$

where  $\mathbf{q}_{123}$ ,  $\mathbf{q}_{12}$ ,  $\mathbf{q}_{23}$ ,  $\mathbf{q}_{31}$ ,  $\mathbf{q}_1$ ,  $\mathbf{q}_2$ ,  $\mathbf{q}_3$  are vectors representing the sets of intersecting velocities and each components free parameters respectively.

These sets are easily identified when referring to Figure 5 (b) and (c). It can be verified that indeed the minimum number of parameters for a particular articulated object is:

$$\sum_{\forall s_o \in \mathcal{O}} \dim(\mathbf{Q}_{s_o}) = 6 + \sum c_j, \quad (31)$$

where  $c_j$  is the class of joint  $j$ .

The mapping  $\mathbf{A}$  is indeed dimension  $(6k) \times (6 + \sum c_j)$ , remembering that  $k$  is the number of components. Figure 5(a) and (b) show that even though the dimensions of the subsets may vary for a same object, the total sum of the dimension of the subspaces remains constant. The reader can refer to [5] for the articulation matrix between two components and one joint.

The generalized parameter vector is therefore:

$$\dot{\mathbf{q}} = \left( \mathbf{q}_{123}, \mathbf{q}_{12}, \mathbf{q}_{23}, \mathbf{q}_{31}, \mathbf{q}_1, \mathbf{q}_2, \mathbf{q}_3 \right). \quad (32)$$

Once these velocities are obtained, they can be related back to the camera frame as in equation (7):

$$\begin{pmatrix} {}^c\mathbf{v}_1 \\ {}^c\mathbf{v}_2 \end{pmatrix} = \mathbf{A} \left( \mathbf{q}_{123}, \mathbf{q}_{12}, \mathbf{q}_{23}, \mathbf{q}_{31}, \mathbf{q}_1, \mathbf{q}_2, \mathbf{q}_3 \right). \quad (33)$$

It is important to highlight in this example the decoupling of the minimization problem. This is apparent in equation (29) where extra zeros appear in the Jacobian compared to the traditional case of a kinematic chain. In the particular case of three components and two joints, a kinematic chain has only 3 zeros in the articulation matrix compared to 9 in the kinematic set case.

## 4 Registration

In this section, a tracking control law is derived for the general class of articulated objects. The aim of the control scheme is to minimize the objective function given in equation (3). Thus, the error function is given as:

$$\begin{pmatrix} \mathbf{e}_1 \\ \vdots \\ \mathbf{e}_k \end{pmatrix} = \mathbf{D} \begin{pmatrix} \mathbf{s}_1(\mathbf{q}) - \mathbf{s}_{d1} \\ \vdots \\ \mathbf{s}_k(\mathbf{q}) - \mathbf{s}_{dk} \end{pmatrix}, \quad (34)$$

where  $\mathbf{q}$  is a vector composed of the minimal set of velocities corresponding to the object's motion and each  $\mathbf{e}_i$  corresponds to an error vector for component  $i$ .  $\mathbf{D}$  is a diagonal weighting matrix corresponding to the likelihood of a particular error within a robust distribution:

$$\mathbf{D} = \begin{pmatrix} \mathbf{D}_1 & & 0 \\ & \ddots & \\ 0 & & \mathbf{D}_k \end{pmatrix},$$

where each matrix  $\mathbf{D}_i$  is a diagonal matrix corresponding to a component which has weights  $w_j$  along the diagonal. These weights correspond to the uncertainty of the measured visual feature  $j$ . The computation of the weights are given in the Appendix 6.

If  $\mathbf{D}$  were constant, the derivative of equation (34) would be given by:

$$\begin{pmatrix} \dot{\mathbf{e}}_1 \\ \vdots \\ \dot{\mathbf{e}}_k \end{pmatrix} = \begin{pmatrix} \frac{\partial \mathbf{e}_1}{\partial \mathbf{s}_1} \frac{\partial \mathbf{s}_1}{\partial \mathbf{r}_1} \frac{\partial \mathbf{r}_1}{\partial \mathbf{q}} \\ \vdots \\ \frac{\partial \mathbf{e}_k}{\partial \mathbf{s}_k} \frac{\partial \mathbf{s}_k}{\partial \mathbf{r}_k} \frac{\partial \mathbf{r}_k}{\partial \mathbf{q}} \end{pmatrix} \dot{\mathbf{q}} = \mathbf{DL}_\mathbf{D}\mathbf{A}\dot{\mathbf{q}}, \quad (35)$$

where  $\dot{\mathbf{q}}$  is the minimal velocity vector,  $\mathbf{A}$  is the articulation matrix describing the mapping in equation (29) and  $\mathbf{L}_\mathbf{D}$  is the block diagonal interaction matrix as in equation (6) given as:

$$\mathbf{L}_\mathbf{D} = \begin{pmatrix} \frac{\partial \mathbf{s}_1}{\partial \mathbf{r}_1} & & \\ & \ddots & \\ & & \frac{\partial \mathbf{s}_k}{\partial \mathbf{r}_k} \end{pmatrix} = \begin{pmatrix} \mathbf{L}_{\mathbf{s}_1} & & \mathbf{0}_6 \\ & \ddots & \\ \mathbf{0}_6 & & \mathbf{L}_{\mathbf{s}_k} \end{pmatrix}. \quad (36)$$

If an exponential decrease of the error  $\mathbf{e}$  is specified:

$$\dot{\mathbf{e}} = -\lambda \mathbf{e}, \quad (37)$$

where  $\lambda$  is a positive scalar, the following control law is obtained by combining equation (37) and equation (35):

$$\dot{\mathbf{q}} = -\lambda(\widehat{\mathbf{D}}\widehat{\mathbf{L}}_\mathbf{D}\widehat{\mathbf{A}})^+\widehat{\mathbf{D}}(\mathbf{s}(\mathbf{q}) - \mathbf{s}_d), \quad (38)$$

where  $\widehat{\mathbf{L}}_\mathbf{D}$  is a model or an approximation of the real matrix  $\mathbf{L}_\mathbf{D}$ .  $\widehat{\mathbf{D}}$  is a chosen model for  $\mathbf{D}$  and  $\widehat{\mathbf{A}}$  depends on the previous estimation of the object configuration  $\mathbf{q}$ . Once these velocities are estimated they can be related back to the camera frame as in equation (7).

If  $\widehat{\mathbf{D}}$ ,  $\widehat{\mathbf{L}}_\mathbf{D}$  and  $\widehat{\mathbf{A}}$  were constant, a sufficient criteria to ensure global asymptotic stability of the system would be given by [27]:

$$(\widehat{\mathbf{D}}\widehat{\mathbf{L}}_\mathbf{D}\widehat{\mathbf{A}})^+\mathbf{DL}_\mathbf{D}\mathbf{A} > 0. \quad (39)$$

As usual, in non-linear minimization, it is impossible to demonstrate the global stability (i.e convergence upon a global minimum). However, it is possible to obtain the local stability (i.e. convergence when the error between subsequent images is small). In the experiments, the current value of the weights, an estimate of the depth at each iteration and the current feature coordinates are used:

$$(\widehat{\mathbf{D}}\widehat{\mathbf{L}}_\mathbf{D}\widehat{\mathbf{A}})^+ = [\mathbf{DL}_\mathbf{D}(\mathbf{s}, \widehat{\mathbf{Z}})\mathbf{A}(\mathbf{s}, \widehat{\mathbf{Z}})]^+. \quad (40)$$

This choice allows the system to follow the intended behavior ( $\dot{\mathbf{e}} = -\lambda \mathbf{e}$ ) as closely as possible. However, even when condition (39) is satisfied, only local stability can be demonstrated since  $\mathbf{L}_\mathbf{D}$  and  $\mathbf{A}$  are not constant (refer to (35) that has been used to derive (39)). A constant Jacobian could be considered using the initial depth  $\mathbf{Z}_i$ , the initial value of the features  $\mathbf{s}_i$  and the first value of the weighting matrix  $\widehat{\mathbf{D}} = \mathbf{I}_m$ , however, experimental results show that the first method is more stable. Of course, it is also necessary to ensure that a sufficient number of features will not be rejected so that  $\mathbf{DL}_\mathbf{D}\mathbf{A}$  is always of full rank ( $m$  for the 3D configuration of an articulated object).

## 5 Results

In this section four experiments are presented for tracking of articulated objects in *real* sequences. The results presented here have been performed using a monocular vision system. Local tracking is performed via a 1D oriented gradient search to the normal of projected contours as detailed further in Section 5.1.

It should be noted that the programming task involved in obtaining these results was not trivial. Indeed it involved modeling the storage of information within an object oriented C++ hierarchy of feature sets and correct implementation of the interaction between these feature sets represented as a graph of feature sets. The basic implementation of the algorithm gives the following pseudo-code:

- 
1. Obtain initial pose.
  2. Capture a new image.
  3. Project the model onto the image.
  4. 1D Search for corresponding points normal to the projected contours.
  5. Determine the distance errors  $\mathbf{e}$  in the image.
  6. Calculate  $(\widehat{\mathbf{D}}\widehat{\mathbf{L}}_s\widehat{\mathbf{D}}\widehat{\mathbf{A}})$ .
  7. Determine minimal subspace velocities as in equation (38).
  8. Update the generalized parameters as in equation (7).
  9. Re-project onto image using each component's pose  $\mathcal{E}$  go to 5 until the error converges.
  10. Repeat to 2.
- 

One may consult our rigid-tracking paper for more detailed information on the implementation of our method [4]. These previous results demonstrate a general method for deriving interaction matrices for any type of distance to contour and also show the robustness of this approach with respect to occlusion and background clutter.

The information included in the following results are aimed at proving the advantages of our articulated tracking method. The experiments test for the following performance factors:

- 3D motion - In order to show that our method is capable of handling all types of 3D movement, both camera and object motion as well as articulated motion have been introduced into each experiment.
- Minimal parameters - Velocity plots show the evolution of the minimal parameter kinematic sets which are used for the tracking. At the same time the velocities attained by our system are given.
- Rank deficient visual information - one of the advantages of our method is that components with little visual information can be tracked stably and robustly since they are now supported by visual information from other components (unlike in the case of the Lagrange Multipliers or individual object tracking). Furthermore, this interdependence of components allows on to avoid badly conditioned Jacobian or noise in the image which would normally lead to failure. This involves using different configurations of distance-to-line type features including simple lines, planes and full 3D objects and comparing with individual object tracking.

- Joint types - linear and non-linear joints have been implemented so as to show the general class of joints which can be defined using our velocity constraint matrix.
- Robust estimation - also enhances the robustness of the method to external occlusion and miss-tracking.
- Precision - the performance of our method is evaluated by looking at the re-projection of the model onto the real scene which presents the data in a visual manner. Furthermore, a plot of the error norm is given, where the ground truth for the error norm is governed by the noise in the image. A pose plot is also given in order to show the large movements imposed within the experiment as well as the smoothness of the estimated pose.

## 5.1 Low-level tracking of visual features

When dealing with low-level image processing, the object model is projected onto the image and the contours are sampled at a regular distance. At these sample points a 1 dimensional search is performed to the normal of the contour for corresponding edges. An *oriented* gradient mask [2] is used to detect the presence of a contour. One of the advantages of this method is that it only searches for edges which are aligned in the same direction as the parent contour. An array of 180 masks is generated off-line which is indexed according to the contour angle. This therefore implemented with convolution efficiency, and leads to real-time performance.

When referring to Figure 6, the process consists of searching for the corresponding point  $p^{t+1}$  in image  $I^{t+1}$  for each point  $p^t$ . A 1D search interval  $\{Q_j, j \in [-J, J]\}$  is determined in the direction  $\delta$  of the normal to the contour. For each position  $Q_j$  lying in the direction  $\delta$  a mask convolution corresponding to the square root of a log-likelihood ratio  $\zeta_j$  is computed. Thus the new position  $p^{t+1}$  is given by:

$$Q^{j*} = \arg \max_{j \in [-J, J]} \zeta_j \text{ with } \zeta_j = | I_{\nu(Q_j)}^{t+1} * M_\delta |$$

$\nu(\cdot)$  is the neighborhood of the considered pixel. In this paper the neighborhood is limited to a  $7 \times 7$  pixel mask. It should be noted that there is a trade-off to be made between real-time performance and mask stability. Likewise there is a trade-off to be made between real-time performance and the maximum inter-frame movement of the object.

This low level search produces a list of  $k$  points which are be used to calculate distances from corresponding projected contours. In this paper, decision thresholds are avoided by propagating low level uncertainty to the global estimation process. The maximum likelihood value is first normalized between 1 and 0. In order to couple the local measure of uncertainty  $\zeta_{j*}$  with the global measure of uncertainty  $w_i$ , the weights are given as:

$$w_{p_i} = \zeta_{j*} w_i, \quad (41)$$

where  $w_{p_i}$  is the propagated weight and the weights  $w_{p_i}$  are used in the matrix  $\mathbf{D}$  with in (38).

This means that the low level edge detection is made simultaneously with the high level robust outlier rejection. This also has the effect of giving the most certainty to strong contours in terms of the local likelihood and amongst those correspondences the M-estimator converges upon those which conform globally to the 3D shape of the object.

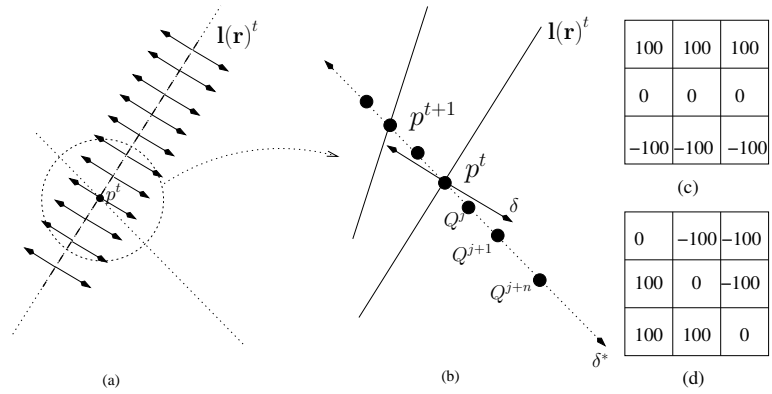


Figure 6: Determining points position in the next image using the oriented gradient algorithm: (a) calculating the normal at sample points, (b) sampling along the normal (c-d) 2 out of 180 3x3 predetermined masks (in practice 7x7 masks are used) (c) 180° (d) 45°.

Effectively the robust estimator chooses which correspondences should be considered instead of a manually chosen threshold. This is advantageous when different scenes are considered along with different size masks.

## 5.2 Rotational Link

This first experiment was carried out for a class 1 type link with a single degree of rotation on the  $x$  axis. Images at different points of one of the test sequences are shown in Figure 7. The constraint vector was defined as in equation (3.3.2) and the object frame was chosen to coincide with the joint frame.

This sequence along with others performed using the same object model have shown real time efficiency with the tracking computation taking on average 25ms per image. Both the hinge and the object were displaced during tracking without failure. The improvement in estimation precision can be observed in the two plots shown in Figure 8. In these plots, individual tracking is compared to articulated tracking. In the case of the hinged door visual information has been removed from the door to demonstrate one of the benefits of the kinematic set approach. In part of this experiment only two parallel lines which are very closely spaced are used as visual information for the door component as seen in (a) and (b). In this case the visual information from the door only constraints 5 degrees of freedom. Furthermore, since the lines are closely spaced, the rotational parameter around the line is also weakly conditioned. Other than this there are also many shadow artifacts and false lines apparent in the sequence. Individual tracking along with methods requiring the pose for each component before applying constraints would fail in this situation. It can be seen that the kinematic set approach succeeds by better distributing the information between components so as to constrain the uncertain parameters of the first component.

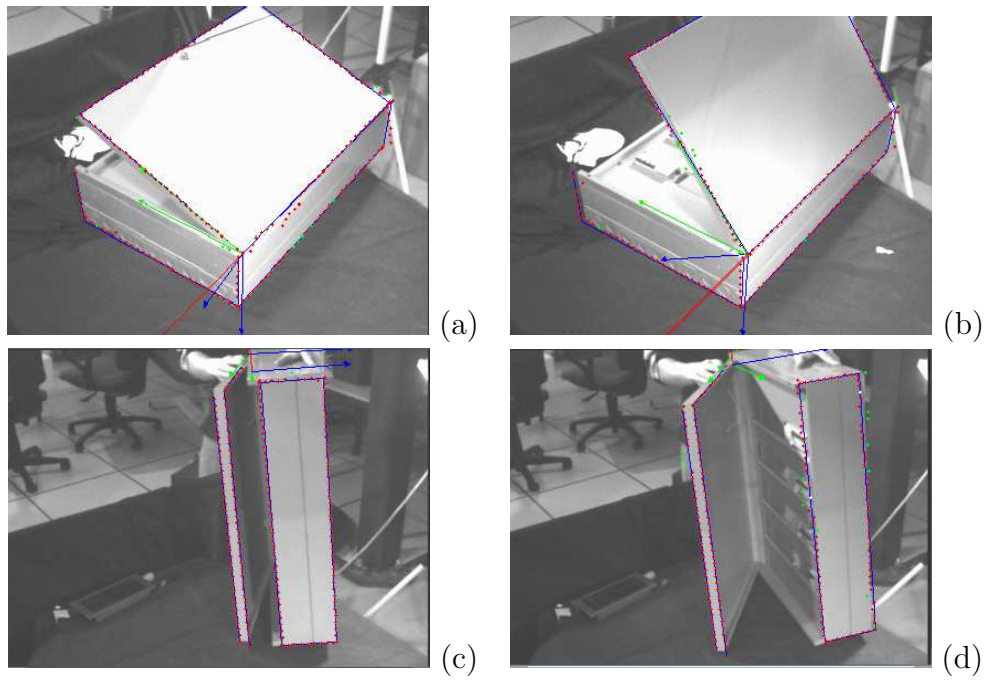


Figure 7: Rotation of a hinged door while undergoing various movements. In this and all the following figures the reference frames for each component are shown in yellow, blue and red for each axis. The projection of the CAD model contour onto the image using the estimated pose parameters is shown in blue. It can be seen that the projected position of the contours are visually acceptable. The points rejected by the M-estimator are shown in green. Changes in illumination on the surface of the door can be seen between (a,b) and (c,d) and partial occlusion by a hand is shown at the top of the box in (c) and (d).

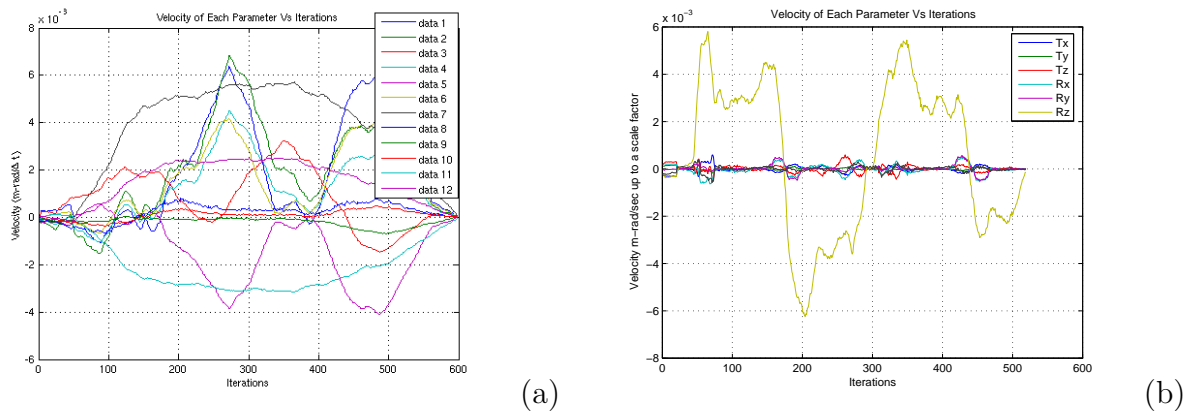


Figure 8: The velocity parameters for tracking of a hinged door in meters-radians per  $\Delta t$ , where  $\Delta t$  is the time for 1 frame capture of the camera. In these experiments frame capture is approximately  $30fps$ , therefore,  $\Delta t = 1/30$ . (a) Objects tracked separately with 12 parameters and tracking fails almost immediately due to severe shadow interference. (b) Articulated object tracking with 7 parameters. Note that only rotational hinge movement was performed for these plots as shown in Figure 7 (a) and (b) as opposed to full object and hinge movement shown in the images in Figure 7 (c) and (d). The opening and closing of the door can be observed in figure (b). The maximum velocity of the door was approximately  $0.15rad/s$

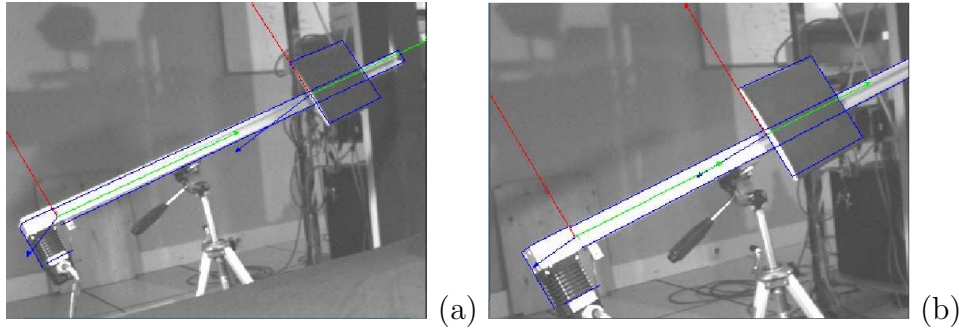


Figure 9: Translation along a sliding mechanism while the camera is moving. The first component is the rail and the second is the square which slides along the rail. It should also be noted that all CAD models were measured very roughly by hand using a simple ruler. Therefore, it can be seen that the tracking is able to support a certain degree of modeling error also.

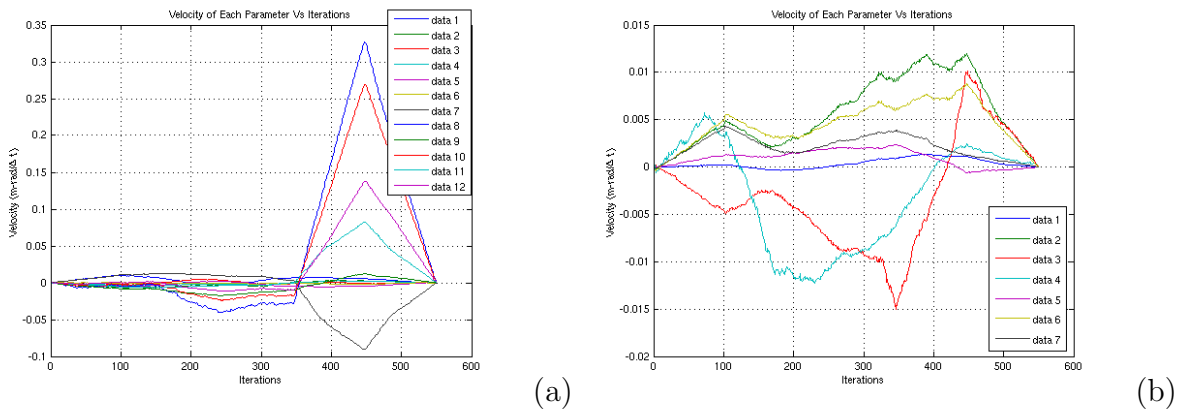


Figure 10: The velocity parameters for a sliding mechanism: (a) Objects tracked separately with 12 parameters and the tracking fails after 350 iterations due to a badly conditioned Jacobian from limited disparity information from the rail. The maximum velocity of the slide was approx.  $0.45m/s$ . (b) Articulated object tracking with 7 parameters. Note that translational movement of the object was performed along with movement of the camera for this experiment.

### 5.3 Prismatic Link

This second experiment was carried out for class one link with a single degree of translation on the  $x$  axis. The constraint vector was defined as a prismatic link along the  $x$  axis:

$$\mathbf{S}_{12}^{\perp} = (1, 0, 0, 0, 0, 0), \quad (42)$$

and the object frame was chosen to coincide with the joint frame.

The sequence, shown in Figure 9, also runs in real time with the tracking computation taking on average 20ms per image. Both the camera and the slide were displaced simultaneously during tracking without failure. It can be seen in the image that the alignment of the drawn contour with the object is visually acceptable. As in the previous experiment, the estimated parameter velocities are compared in Figure 10. It can be seen that tracking fails in 10(a) at around 350 iterations when individual tracking is used and in 10(b) tracking is successful.

## 5.4 Helical Link

This experiment, reported in Figure 11, was carried out for class one link with helical movement simultaneously along and around the  $z$  axis. This type of link is more complex due to the nonlinear articulation matrix mapping from the different component twists to the minimal subspaces. The constraint vector was defined as in equation (3.3.2) and the object frame was chosen to coincide with the joint frame. Note that the constraint vector was defined by taking into consideration that for  $10 \times 2\pi$  rotations of the screw, it translated 4.5 centimeters along the  $z$  axis.

Tracking of this object also ran in real-time with the main loop computation taking on average 25ms per image. It should be noted that although the presented sequence works for individual tracking in this particular sequence, it often fails completely due to the limited contour information, background noise or occlusions. A working case is shown in Figure 12(a) for comparative purposes. When tracked simultaneously with the plate, as an articulated object, the tracking of the screw is also based on the measurements of the plate making the tracking more stable and computationally efficient. It can be seen that tracking continuous throughout a long sequence of approximately 1100 images.

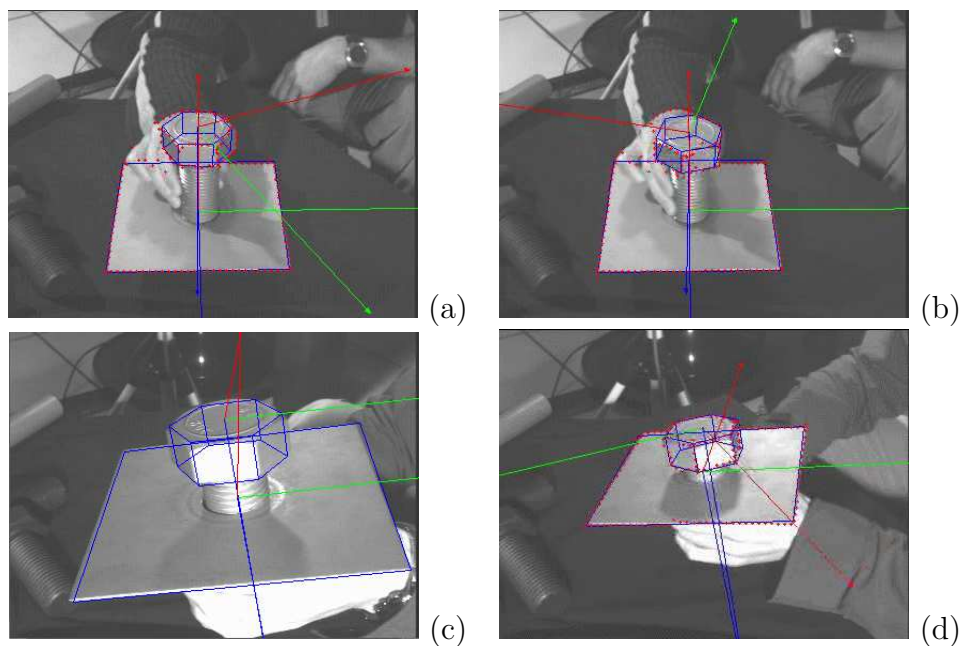


Figure 11: Helical movement of a screw while the screw and the platform are simultaneously in movement. Partial occlusion is made by a hand and a wide range of positions and configurations are tested. The object is moved and manipulated by hand and smooth movements of the projected model are observed without the need for filtering.

## 5.5 Robotic Arm

An experiment was carried out for two class one links on a robotic arm so as to generalize the approach to 3 or more joints. The articulations tracked were rotational links around the  $z$  and  $x$  axes. The constraint vectors were each defined by a pose and a constraint

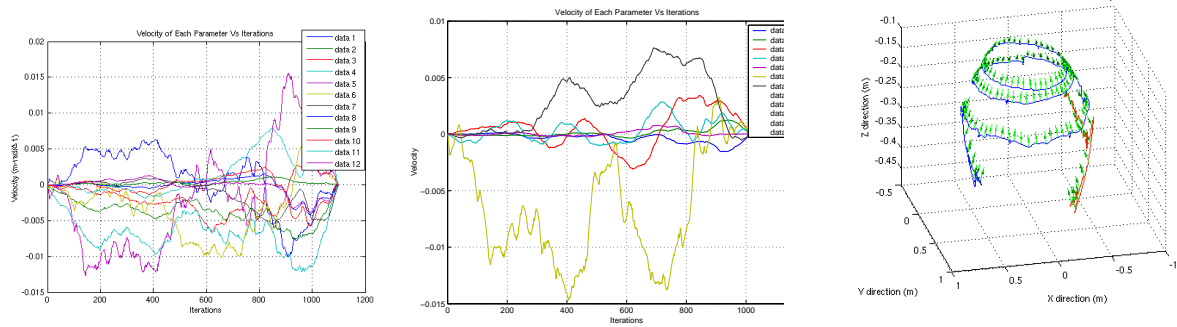


Figure 12: The velocity parameters for helical movement of a screw. (a) Objects tracked separately with 12 parameters. (b) Articulated object tracking with 7 parameters. (c) The pose of both the screw (in blue) and the plate (in red) of the components wrt. the camera. Note that helical movement of the screw was performed by turning the screw and the object was also moved during these experiments. The helical movement is in green in figure (b). Any jitter observed in the pose can be attributed to hand jitter. It can be seen that only one parameter more than required for a rigid object is needed to estimate the helical movement along and around the  $x$  axis.

matrix as given in equation (3.3.2). Note that the constraints are no longer defined in the same coordinate system as in the previous case.

This sequence also displays real time efficiency with the tracking computation taking on average 25ms per image. As mentioned in the caption of Figure 14, only two line features are used to track on the forearm components and are therefore not full rank for rigid pose calculation. Consequently the rigid pose plots could not be compared. As in the previous experiment, the kinematic set formulation is able to handle tracking this situation.

In what follows, a table comparing and showing the improvement in accuracy for tracking multi-body objects is given.

type / mm	Hinge	Slide	Screw	Robot
Individual	miss-track	1 badly conditioned	2.2	Rank deficient
Articulated	2.2	0.8	2	2.2

Table 4: A comparison of the error norm between individual tracking methods and articulated tracking via kinematic sets

When referring to the table in Table 4, it can be mentioned that for the hinge, tracking failure is more easily observed in the image as the tracking continues without fully breaking down in the plots. Even so, the plot of the velocities shows different values to that of the articulated case. This failure was due to a shadow on the box which was better defined than a contour on the object. The miss-tracking due to this shadow broke down the estimation gradually in the individual case. In the kinematic set case the miss-tracking did not disturb the pose estimation because more information from the other component balanced out these errors. In the case of the slide, there was not enough visual information from the rail to constrain a 6 parameter pose. Indeed the object model is essentially composed of two parallel lines in close proximity. Once again

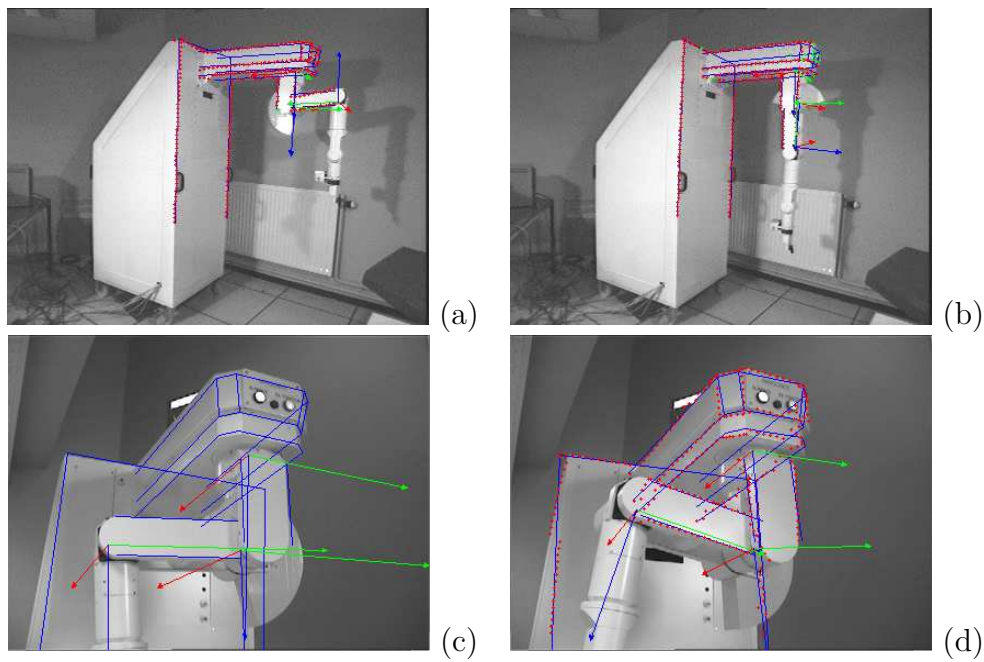


Figure 13: Movement of a robotic arm with two degrees of freedom. In this example, it can be seen that this approach generalizes to more than one parameter. In (a) and (b) it can be seen that even when very little visual information is available, the estimation of unknown joint parameters can be achieved by relying on the more stable estimation obtained from the base of the robot. Various positions and configurations have been tested.

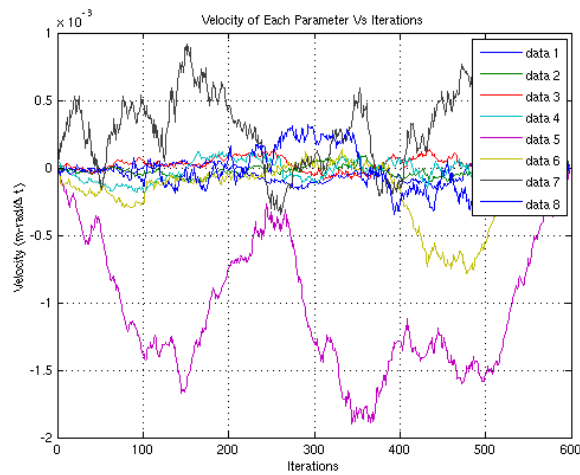


Figure 14: The estimated velocity parameters for tracking of a robot arm with 8 parameters. Note that two fore-arm components are rank deficient so it was not possible to estimate their pose individually.

the kinematic set approach was able to draw on the other component to fully constrain the tracking. In the screw case, the background is essentially black against a silver object making strong visual features. Furthermore, each component is well constrained individually, therefore individual tracking succeeds, however, the computation time is twice that of the articulated case. Finally, the case of the robot again demonstrates the advantage of our approach within complex scenarios and rank deficient systems.

## 6 Conclusion

The method presented here demonstrates an efficient and accurate approach to real-time tracking of complex articulated objects. A framework is given for defining any type of mechanical link between components of an object. A method for object-based tracking has been derived and implemented. Furthermore, a kinematic set formulation for tracking articulated objects has been described. It has been shown that it is possible to decouple the interaction between articulated components using this approach. Subsequent computational efficiency and visual precision have been demonstrated.

In perspective, it would be interesting to explore the possibility of treating closed link systems using kinematic sets for both visual servoing and tracking applications. Elastic and other sorts of physical constraints equations could be introduced into the constraint matrix. This formalism could also be used in various applications such as passively constraining medical robot manipulators for surgical tasks or pose calculation and interaction parameter estimation for augmented reality based human computer interaction.

## Appendix

### Computing confidence

This section gives a brief overview for the calculation of weights for each image feature. The weights  $w_i$ , which represent the different elements of the  $\mathbf{D}$  matrix and reflect the confidence of each feature, are usually given by [14]:

$$w_i = \frac{\psi(\delta_i/\sigma)}{\delta_i/\sigma}, \quad (43)$$

where  $\psi(u) = \frac{\partial \rho(u)}{\partial u}$  ( $\psi$  is the influence function) and  $\delta_i$  is the normalized residue given by  $\delta_i = \Delta_i - Med(\Delta)$  (where  $Med(\Delta)$  is the median operator).

Of the various loss and corresponding influence functions that exist in the literature Tukey's hard re-descending function is considered. Tukey's function completely rejects outliers and gives them a zero weight. This is of interest in tracking applications so that a detected outlier has no effect on the virtual camera motion. This influence function is given by:

$$\psi(u) = \begin{cases} u(C^2 - u^2)^2 & , \text{ if } |u| \leq C \\ 0 & , \text{ else,} \end{cases} \quad (44)$$

where the proportionality factor for Tukey's function is  $C = 4.6851$  and represents 95% efficiency in the case of Gaussian Noise.

In order to obtain a robust objective function, a value describing the certainty of the measures is required. The scale  $\sigma$  or the estimated standard deviation of the inlier data and is an important value for the efficiency of the method. Since scale varies significantly during convergence we choose to estimate it online using the Median Absolute Deviation (MAD):

$$\hat{\sigma} = \frac{1}{\Phi^{-1}(0.75)} \text{Med}_i(|\delta_i - \text{Med}_j(\delta_j)|). \quad (45)$$

where  $\Phi()$  is the cumulative normal distribution function and  $\frac{1}{\Phi^{-1}(0.75)} = 1.48$  represents one standard deviation of the normal distribution.

The introduction of the weighting matrix  $\mathbf{D}$  into the minimization scheme in Section 4 is achieved via an iteratively re-weighted least squares implementation. Robust weights were calculated together for each component's feature set due to incompatibilities when calculating weights directly from all the object features.

## References

- [1] J.K. Aggarwal, Q. Cai, W. Liao, and B. Sabata. Nonrigid motion analysis: Articulated and elastic motion. *Computer Vision and Image Understanding*, 70(2):142–156, 1998.
- [2] P. Bouthemy. A maximum likelihood framework for determining moving edges. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 11(5):499–511, 1989.
- [3] R. W. Brockett. Robotic manipulators and the product of exponentials formula. In *Mathematical Theory of Networks and Systems*, pages 120–29, 1984.
- [4] A.-I. Comport, E. Marchand, and F. Chaumette. A real-time tracker for markerless augmented reality. In *ACM/IEEE Int. Symp. on Mixed and Augmented Reality, ISMAR'03*, pages 36–45, Tokyo, Japan, 2003.
- [5] A.-I. Comport, E. Marchand, and F. Chaumette. Object-based visual 3d tracking of articulated objects via kinematic sets. In *IEEE Workshop on Articulated and Non-Rigid Motion*, Washington, DC, 2004.
- [6] Q. Delamarre and O. Faugeras. 3d articulated models and multi-view tracking with silhouettes. In *International Conference on Computer Vision*, 1999.
- [7] D. Dementhon and L. Davis. Model-based object pose in 25 lines of codes. *Int. J. of Computer Vision*, 15:123–141, 1995.
- [8] J. Denavit and R. S. Hartenberg. A kinematic notation for lower-pair mechanisms based on matrices. *J. Appl. Mechanics*, June 1955, 22:215–221, 1955.
- [9] Jonathan Deutscher and Ian Reid. Articulated body motion capture by stochastic search. *International Journal of Computer Vision*, 61(2):185–205, 2005.
- [10] M. Dhome, M. Richetin, J.-T. Lapresté, and G. Rives. Determination of the attitude of 3-d objects from a single perspective view. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 11(12):1265–1278, 1989.
- [11] T. Drummond and R. Cipolla. Real-time visual tracking of complex structures. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 27(7):932–946, 2002.

- [12] B. Espiau, F. Chaumette, and P. Rives. A new approach to visual servoing in robotics. *IEEE Trans. on Robotics and Automation*, 8(3):313–326, 1992.
- [13] P. Fua and C. Brechbuhler. Imposing hard constraints on deformable models through optimization in orthogonal subspaces. *Computer Vision and Image Understanding*, 65(2):148–162, 1997.
- [14] P.-J. Huber. *Robust Statistics*. Wiler, New York, 1981.
- [15] M. Isard and A. Blake. Contour tracking by stochastic propagation of conditional density. In *European Conference on Computer Vision, ECCV'96, LNCS no. 1064, Springer-Verlag*, pages 343–356, Cambridge, UK, 1996.
- [16] David G. Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 60(2):91–110, 2004. TY - JOUR.
- [17] D.G. Lowe. Three-dimensional object recognition from single two-dimensional images. *Artificial Intelligence*, 31(3):355–394, 1987.
- [18] D.G. Lowe. Fitting parameterized three-dimensional models to images. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 13(5):441–450, 1991.
- [19] C.P. Lu, G.D. Hager, and E. Mjolsness. Fast and globally convergent pose estimation from video images. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 22(6):610–622, 2000.
- [20] E. Marchand and F. Chaumette. Virtual visual servoing: a framework for real-time augmented reality. In *EUROGRAPHICS'02 Conference Proceeding*, volume 21(3) of *Computer Graphics Forum*, pages 289–298, Saarebrücken, Germany, 2002.
- [21] D.N. Metaxas and I.A. Kakadiaris. Elastically adaptive deformable models. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 24(10):1310–1321, 2002. TY - JOUR.
- [22] R.M. Murray, L. Zexiang, and S.S Sastry. *A Mathematical introduction to robotic manipulation*. CRC Press, 1994.
- [23] T. Nunomaki, S. Yonemoto, D. Arita, and R. Taniguchi. Multipart non-rigid object tracking based on time model-space gradients. In *First International Workshop on Articulated Motion and Deformable Objects*, pages 78–82, 2000.
- [24] R. Plankers and P. Fua. Articulated soft objects for multiview shape and motion capture. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 25(9):1182–1187, 2003.
- [25] R.M Rosenberg. *Analytical Dynamics of Discrete Systems*. Plenum Press, New York, 1977.
- [26] A. Ruf and R. Horaud. Rigid and articulated motion seen with an uncalibrated stereo rig. In *IEEE Int. Conf. on Computer Vision*, volume 2, pages 789–796, Corfu, Greece, 1999.
- [27] C. Samson, M. Le Borgne, and B. Espiau. *Robot Control: the Task Function Approach*. Clarendon Press, Oxford, United Kingdom, 1991.
- [28] Ying Wu, Gang Hua, and Ting Yu. Tracking articulated body by dynamic markov network. In *Computer Vision, 2003. Proceedings. Ninth IEEE International Conference on*, pages 1094–1101 vol.2, 2003.

- [29] Lin Zhou, C. Kambhamettu, D.B. Goldgof, K. Palaniappan, and A.F. Hasler. Tracking non-rigid motion and structure from 2d satellite cloud images without correspondences. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 23(11):1330–1336, 2001.