



## Pose Estimation in Conformal Geometric Algebra Part II: Real-Time Pose Estimation Using Extended Feature Concepts

BODO ROSENHAHN AND GERALD SOMMER

*Cognitive Systems Group, Institute of Computer Science and Applied Mathematics,  
Christian-Albrechts-University of Kiel, D-24105 Kiel, Germany*

bro@ks.informatik.uni-kiel.de

gs@ks.informatik.uni-kiel.de

**Abstract.** Part II uses the foundations of Part I [35] to define constraint equations for 2D-3D pose estimation of different corresponding entities. Most articles on pose estimation concentrate on specific types of correspondences, mostly between points, and only rarely use line correspondences. The first aim of this part is to extend pose estimation scenarios to correspondences of an extended set of geometric entities. In this context we are interested to relate the following (2D) image and (3D) model types: 2D point/3D point, 2D line/3D point, 2D line/3D line, 2D conic/3D circle, 2D conic/3D sphere. Furthermore, to handle articulated objects, we describe kinematic chains in this context in a similar manner. We ensure that all constraint equations end up in a distance measure in the Euclidean space, which is well posed in the context of noisy data. We also discuss the numerical estimation of the pose. We propose to use linearized twist transformations which result in well conditioned and fast solvable systems of equations. The key idea is not to search for the representation of the Lie group, describing the rigid body motion, but for the representation of their generating Lie algebra. This leads to real-time capable algorithms.

**Keywords:** 2D-3D pose estimation, pose constraints, kinematic chains, circles, spheres, twists

### 1. Introduction

This contribution concerns the 2D-3D pose estimation problem of different corresponding entities. Pose estimation itself is one of the oldest computer vision problems and algebraic solutions with different camera models have been proposed for several variations of this problem. Pioneering work was done in the 80's and 90's by Lowe [22, 23], Grimson [12] and others. In their work point correspondences are used. Other works concerning lines or line segments can be found in e.g. [19, 42]. Works concerning extensions to kinematic chains can be found in [4, 13]. Nearly all papers concentrate on one specific type of correspondences. But many situations are conceivable in which a system has to gather information from different hints or has to consider different reliabilities of measurements. This is the main aspect of this work: We describe a scenario for adaptive pose estimation of simultaneously used different enti-

ties, without losing linearity, well conditioned equations and real-time capability. This work contains the second part of our research on pose estimation. The first part [35] discussed the scenario and the mathematical foundations for the pose problem. This part uses these foundations to deal with the pose estimation problem.

Section 2 starts with the pose constraints to relate 3D point, line and plane features. In Section 3 we extend this formalization to kinematic chains, 3D circles and 3D spheres. The aim is to model all different kinds of entities, their transformations and relations in one algebraic framework to get constraint equations which can be used in a similar manner and can be solved simultaneously. Furthermore, we explain how to use the constraints in a noise adaptive way. This means, we control the influence of a constraint to the whole system of equations. This is only possible if the constraints describe (in their implicit formulations) equivalent geometries. In this context every constraint results in a

Hesse distance error measure between the involved entities. In the experimental part, Section 4, we present several experiments which visualize the properties of our algorithms.

### 1.1. Notations

This part uses the foundations of Part I [35] to deal with the 2D-3D pose estimation problem in a geometric context. The language we use are Clifford or geometric algebras [38]. Euclidean, projective and conformal geometry find in geometric algebra the frame where they can reconcile and express their potential. This opens for example a new alternative for the mathematical treatment of the stratification of the 3D visual space. Therefore we will first repeat the notations introduced in Part I so that the reader can more easily follow this work.

The term  $\mathcal{G}_{p,q,r}$  denotes a geometric algebra with  $p$ ,  $q$  and  $r$  basis vectors which square to  $+1$ ,  $-1$ , and  $0$  respectively. We further use  $\mathcal{G}_3$  to model 3D Euclidean geometry,  $\mathcal{G}_{3,1}$  to model 3D projective geometry and  $\mathcal{G}_{4,1}$  to model 3D conformal geometry. Multivectors are written boldface and the geometric product of two multivectors  $\mathbf{A}$  and  $\mathbf{B}$  is denoted by juxtaposition,  $\mathbf{AB}$ . Special products defined on the geometric product are the outer ( $\wedge$ ), inner ( $\cdot$ ), commutator ( $\underline{\times}$ ) and anticommutator ( $\overline{\times}$ ) product. The dual of an entity  $\mathbf{X}$  is written as  $\mathbf{X}^*$  and the reverse of a blade  $\mathbf{A}$  as  $\tilde{\mathbf{A}}$ . The join of two blades  $\mathbf{A}$  and  $\mathbf{B}$  is denoted as  $\mathbf{A} \hat{\wedge} \mathbf{B}$  and the meet as  $\mathbf{A} \vee \mathbf{B}$ . 3D Euclidean entities are written in small boldface letters, e.g. a point  $\mathbf{x} \in \mathcal{G}_3$ . Projective entities are written in big boldface letters, e.g. a point  $\mathbf{X} = \mathbf{x} + \mathbf{e}_- \in \mathcal{G}_{3,1}$  and conformal entities are written big boldface underlined, e.g.  $\underline{\mathbf{X}} = \mathbf{x} + \frac{1}{2}\mathbf{x}^2\mathbf{e} + \mathbf{e}_0$ . Rigid motions are handled as screw transformations. A motor  $\mathbf{M} = \exp(-\frac{\theta}{2}(\mathbf{l} + \mathbf{em}))$  formalizes a screw motion. A conformal entity, e.g. a point  $\underline{\mathbf{X}}$ , can be transformed in space by calculating  $\underline{\mathbf{X}}' = \mathbf{M}\underline{\mathbf{X}}\tilde{\mathbf{M}}$ .

## 2. Collinearity and Coplanarity Constraints in Conformal Geometric Algebra

So far we have introduced (see Part I) the representation of entities, their transformations and the interaction of entities between Euclidean, projective and conformal geometry. The aim is now to formalize the pose estimation problem in an implicit way using a set of geometric constraints which describe an error measure to be minimized.

Note: As can be seen from Part I [35], the transformation of an entity given in the projective geometric algebra to the conformal geometric algebra always leads to a dual representation of the entity since

$$\begin{aligned} \mathbf{e} \wedge \mathbf{X} &= \underline{\mathbf{X}}^* \\ \mathbf{e} \wedge \mathbf{L} &= \underline{\mathbf{L}}^* \\ \mathbf{e} \wedge \mathbf{P} &= \underline{\mathbf{P}}^*. \end{aligned} \quad (2.1)$$

In the next sections we will only work in the dual representation of the entities and therefore, from now on we will neglect the  $\star$ -sign in the equations.

In this section, we will derive constraints for collinearity and coplanarity to relate points, lines and planes. The constraints will be given in the conformal space. They are then translated in an error measure of the Euclidean space. While this section only concerns the relation of points, lines and planes, the following sections will regard the constraints to relate the other entities as circles and spheres.

Table 1 gives an overview of the formulations of the constraints for collinearity and coplanarity of points, lines and planes in conformal geometric algebra. Indeed there is no unique representation to model incidence of entities. Therefore we searched for an expression of collinearity and coplanarity which is not only compact and linear, but also contains a geometric distance measure which can numerically stable and fast applied to the pose problem. The reason why we use these equations is the fact, that they express such a geometric distance measure without introducing nonlinearities within the unknowns. The constraints are inspired by Blaschke [3] who formalized three constraint equations for incidence of points, lines and planes in the dual-quaternions. We then translated the equations in [32, 39] to the motor algebra and the conformal geometric algebra, respectively. So far we found no better equations for our scenario which are compact, linear, contain a geometric distance measure, can be applied to a perspective camera model and are suited for the use of different entities simultaneously.

Table 1. The geometric constraints for collinearity and coplanarity of points, lines and planes expressed in conformal geometric algebra.

| Entities    | Constraint in conformal geometric algebra                  |
|-------------|--|
| Point-line  | $\underline{\mathbf{X}} \times \underline{\mathbf{L}} = 0$ |
| Point-plane | $\underline{\mathbf{X}} \times \underline{\mathbf{P}} = 0$ |
| Line-plane  | $\underline{\mathbf{L}} \times \underline{\mathbf{P}} = 0$ |

Now we will analyze the geometry of the constraints introduced in table 1.

### 2.1. Point-Line Constraint

Evaluating the point-line-constraint of a point  $\underline{X} \in \mathcal{G}_{4,1}$ ,  $\underline{X} = \underline{E} + \underline{e}x$ , collinear with a line  $\underline{L} \in \mathcal{G}_{4,1}$ ,  $\underline{L} = \underline{E}r + \underline{e}m$  leads to

$$\begin{aligned}
 0 &= \underline{X} \times \underline{L} \\
 &= \frac{1}{2}(\underline{XL} - \underline{LX}) \\
 &= \frac{1}{2}((\underline{E} + \underline{e}x)(\underline{E}r + \underline{e}m) \\
 &\quad - (\underline{E}r + \underline{e}m)(\underline{E} + \underline{e}x)) \\
 &= \frac{1}{2}(\underline{e}x\underline{E}r + \underline{E}e\underline{m} + \underline{E}^2r - \underline{e}m\underline{E} - r\underline{E}e\underline{x} - r) \\
 &= \frac{1}{2}(\underline{e}xr - \underline{m}e - \underline{m}e - r\underline{x}e) \\
 &= \frac{1}{2}(-2\underline{m} - (xr - rx))\underline{e} \\
 &= -(\underline{m} - \underline{x} \times \underline{r})\underline{e}
 \end{aligned}$$

$$\Leftrightarrow 0 = (\underline{m} - \underline{x} \times \underline{r})\underline{e} \cdot \underline{e}_+ = \underline{m} - \underline{x} \times \underline{r} \quad (2.2)$$

The product  $\underline{X} \times \underline{L}$  is an element of the null space, since  $\underline{e}^2 = 0$ . By calculating the inner product with  $\underline{e}_+$ , we can change this expression to an equation in the non-null space. Note, that this is consistent with Table 6 of Part I. Estimating the inner product with  $\underline{e}_+$  leads to an error direction in the projective space and since the homogeneous component is zero it is simultaneously a vector expression in  $\mathcal{G}_{3,0}$ , the algebra of the 3D Euclidean space.

The term  $\underline{m} - \underline{x} \times \underline{r}$  means that the moment  $\underline{m}$  of a line, which is generated by the outer product of the direction  $\underline{r}$  of the line with a point  $\underline{x}$  on the line, is independent of the chosen point of the line. This is a clear fact from Plücker representation of lines [3].

So far the constraint equation is given unscaled. Following Part I, we have to apply a scaling parameter  $\lambda \in \mathbb{R}$  to express a distance measure in the Euclidean space. In this case let  $\lambda = \frac{1}{\|\underline{r}\|}$ . That means we scale the equation with the inverse norm of the direction of the line. Then we get

$$\begin{aligned}
 0 &= \underline{m} - \underline{x} \times \underline{r} \\
 \Leftrightarrow 0 &= \lambda(\underline{m} - \underline{x} \times \underline{r}) \\
 \Leftrightarrow 0 &= \lambda\underline{m} - \underline{x} \times (\lambda\underline{r}) \\
 \Leftrightarrow 0 &= \underline{m}' - \underline{x} \times \underline{r}'. \quad (2.3)
 \end{aligned}$$

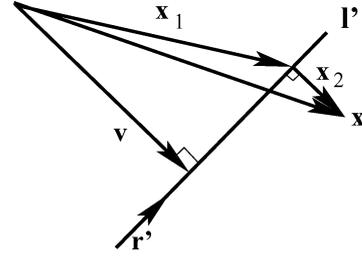


Figure 1. The Euclidean line  $l'$  consists of the direction  $r'$  and the moment  $m' = v \times r'$ . Further, there exists a decomposition  $x = x_1 + x_2$  with  $x_1 \in l'$  and  $x_2 \perp r'$  so that  $m' = v \times r' = x_1 \times r'$ .

The aim is to analyze the bivector  $\underline{m}' - \underline{x} \times \underline{r}'$ . Suppose  $\underline{X} \notin \underline{L}'$ . Then, nonetheless, there exists a decomposition  $x = x_1 + x_2$  with  $\underline{X}_1 \in \underline{L}'$ ,  $\underline{X}_1 = (\underline{E} + \underline{e}x_1)$  and  $\underline{X}_2 \perp \underline{L}'$ ,  $\underline{X}_2 = (\underline{E} + \underline{e}x_2)$ . Figure 1 shows the scenario. Then we can calculate

$$\begin{aligned}
 \|\underline{m}' - \underline{x} \times \underline{r}'\| &= \|\underline{m}' - (x_1 + x_2) \times \underline{r}'\| \\
 &= \|\underline{m}' - x_1 \times \underline{r}' - x_2 \times \underline{r}'\| \\
 &= \|x_2 \times \underline{r}'\| = \|x_2\|. \quad (2.4)
 \end{aligned}$$

Thus, satisfying the scaled point-line constraint means to equate the bivectors  $\underline{m}'$  and  $\underline{x} \times \underline{r}'$ , respectively making the Hesse distance  $\|x_2\|$  of the Euclidean point  $x$  to the Euclidean line  $l'$  to zero.

### 2.2. Point-Plane Constraint

Evaluating the point-plane-constraint of a point  $\underline{X} \in \mathcal{G}_{4,1}$ ,  $\underline{X} = \underline{E} + \underline{e}x$ , coplanar to a plane  $\underline{P} \in \mathcal{G}_{4,1}$ ,  $\underline{P} = \underline{E}n + \underline{e}d\underline{I}_E$ , leads to<sup>1</sup>

$$\begin{aligned}
 0 &= \underline{X} \times \underline{P} \\
 &= \frac{1}{2}(\underline{XP} - \underline{PX}) \\
 &= \frac{1}{2}((\underline{E} + \underline{e}x)(\underline{E}n + \underline{e}d\underline{I}_E) - (\underline{E}n + \underline{e}d\underline{I}_E) \\
 &\quad \times (\underline{E} + \underline{e}x)) \\
 &= \frac{1}{2}(\underline{e}x\underline{E}n + \underline{E}e\underline{d}\underline{I}_E + \underline{n} - \underline{e}d\underline{I}_E\underline{E} - \underline{E}n\underline{e}x - \underline{n}) \\
 &= \frac{1}{2}(-\underline{x}n\underline{e} + d\underline{I}_E\underline{e} + \underline{n} + d\underline{I}_E\underline{e} - \underline{n}x\underline{e} - \underline{n}) \\
 &= \frac{1}{2}(2d\underline{I}_E - (\underline{x}n + \underline{n}x))\underline{e} \\
 &= (d\underline{I}_E - \underline{x} \overline{\times} \underline{n})\underline{e} \\
 \Leftrightarrow 0 &= (d\underline{I}_E - (\underline{x} \overline{\times} \underline{n}))\underline{e} \cdot \underline{e}_+ = d\underline{I}_E - (\underline{x} \overline{\times} \underline{n}). \quad (2.5)
 \end{aligned}$$

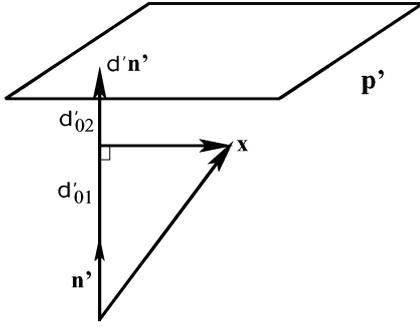


Figure 2. The Euclidean plane  $\underline{p}'$  is represented by the normal  $\underline{n}'$  (as bivector) and the Hesse distance  $d'$ . The value  $d'$  can be interpreted as a sum  $d' = d'_{01} + d'_{02}$  so that  $d'_{01}\underline{n}'$  corresponds to the orthogonal projection of  $\underline{x}$  onto  $\underline{n}'$ .

Note here that the anticommutator product of the bivector  $\underline{n}$  and the vector  $\underline{x}$  results in a trivector, which is subtracted from  $d\underline{I}_E$ . Again the constraint equation is given in the null space which is then transformed to the non-null space by calculating the dot-product with  $\underline{e}_+$ . This leads directly to a scalar value as element of the Euclidean geometric algebra. To express a distance measure in the Euclidean space, let  $\lambda = \frac{1}{\|\underline{n}\|}$ , the inverse of the norm of the bivector  $\underline{n}$ . Then we get

$$\begin{aligned} 0 &= d\underline{I}_E - (\underline{x} \overline{\times} \underline{n}) \\ \Leftrightarrow 0 &= \lambda(d\underline{I}_E - \underline{x} \overline{\times} \underline{n}) \\ \Leftrightarrow 0 &= \lambda d\underline{I}_E - \underline{x} \overline{\times} (\lambda \underline{n}) \\ \Leftrightarrow 0 &= d'\underline{I}_E - \underline{x} \overline{\times} \underline{n}'. \end{aligned} \quad (2.6)$$

Suppose  $\underline{X} \notin \underline{P}'$ . The value  $d'$  can be interpreted as the sum of distances, so that  $d' = d'_{01} + d'_{02}$  and  $d'_{01}\underline{n}'$  is the orthogonal projection of  $\underline{x}$  onto  $\underline{n}'$ . Figure 2 shows the scenario. Then we can calculate

$$\begin{aligned} d'\underline{I}_E - \underline{x} \overline{\times} \underline{n}' &= (d'_{01} + d'_{02})\underline{I}_E - \underline{x} \overline{\times} \underline{n}' \\ &= d'_{02}\underline{I}_E. \end{aligned} \quad (2.7)$$

The value of the expression  $d'\underline{I}_E - \underline{x} \overline{\times} \underline{n}'$  corresponds to the Hesse distance of the Euclidean point  $\underline{x}$  to the Euclidean plane  $\underline{p}'$ .

### 2.3. Line-Plane Constraint

Evaluating the line-plane-constraint of a line  $\underline{L} \in \mathcal{G}_{4,1}$ ,  $\underline{L} = \underline{E}\underline{r} + \underline{e}\underline{m}$ , coplanar to a plane  $\underline{P} \in \mathcal{G}_{4,1}$ ,  $\underline{P} =$

$\underline{E}\underline{n} + \underline{e}\underline{I}_E d$ , leads to

$$\begin{aligned} 0 &= \underline{L} \overline{\times} \underline{P} \\ &= \frac{1}{2}(\underline{L}\underline{P} + \underline{P}\underline{L}) \\ &= \frac{1}{2}((\underline{E}\underline{r} + \underline{e}\underline{m})(\underline{E}\underline{n} + \underline{e}d\underline{I}_E) + (\underline{E}\underline{n} + \underline{e}d\underline{I}_E) \\ &\quad \times (\underline{E}\underline{r} + \underline{e}\underline{m})) \\ &= \frac{1}{2}(\underline{e}\underline{m}\underline{E}\underline{n} + \underline{r}\underline{E}\underline{e}\underline{I}_E d + \underline{r}\underline{n} + \underline{e}\underline{I}_E d \underline{r}\underline{E} + \underline{E}\underline{n}\underline{e}\underline{m} \\ &\quad + \underline{E}\underline{n}\underline{r}\underline{E}) \\ &= \frac{1}{2}(\underline{m}\underline{n}\underline{e} + \underline{r}\underline{I}_E d \underline{e} + \underline{r}\underline{n} + \underline{I}_E d \underline{r}\underline{e} - \underline{n}\underline{m}\underline{e} + \underline{n}\underline{r}) \\ &= \frac{1}{2}((\underline{r}\underline{n} + \underline{n}\underline{r}) + (2\underline{r}\underline{I}_E d + \underline{m}\underline{n} - \underline{n}\underline{m})\underline{e}) \\ &= \frac{1}{2}((\underline{r}\underline{n} + \underline{n}\underline{r}) + 2(\underline{r}\underline{I}_E d + \underline{m} \underline{\times} \underline{n})\underline{e}) \\ &= \underline{r} \overline{\times} \underline{n} + (\underline{r}\underline{I}_E d + \underline{m} \underline{\times} \underline{n})\underline{e} \end{aligned} \quad (2.8)$$

Thus, the constraint of coplanarity of a line to a plane can be partitioned into a constraint on the non-null part of the motor and a constraint on the null part of the motor. This can directly seen in Eq. (2.8) since  $\underline{e}^2 = 0$ .

Again the constraint equation is given unscaled. Let be  $\lambda = \frac{1}{\|\underline{n}\|\|\underline{r}\|}$ . Then we get

$$\begin{aligned} 0 &= \underline{r} \overline{\times} \underline{n} + (\underline{r}\underline{I}_E d + \underline{m} \underline{\times} \underline{n})\underline{e} \\ \Leftrightarrow 0 &= \lambda(\underline{r} \overline{\times} \underline{n} + (\underline{r}\underline{I}_E d + \underline{m} \underline{\times} \underline{n})\underline{e}) \\ \Leftrightarrow 0 &= \underline{r}' \overline{\times} \underline{n}' + (\underline{r}'\underline{I}_E d' + \underline{m}' \underline{\times} \underline{n}')\underline{e}, \end{aligned} \quad (2.9)$$

with

$$\underline{r}' = \frac{1}{\|\underline{r}\|}\underline{r} \quad \underline{n}' = \frac{1}{\|\underline{n}\|}\underline{n} \quad \underline{m}' = \frac{1}{\|\underline{r}\|}\underline{m} \quad d' = \frac{1}{\|\underline{n}\|}d.$$

Suppose  $\underline{L}' \notin \underline{P}'$ . If  $\underline{r}' \not\perp \underline{n}'^*$ , the non-null part leads to

$$\underline{r}' \overline{\times} \underline{n}' = \|\underline{r}'\|\|\underline{n}'\| \cos(\alpha) = \cos(\alpha), \quad (2.10)$$

where  $\alpha$  is the angle between  $\underline{L}'$  and  $\underline{P}'$ , see Fig. 3. If  $\underline{r}' \perp \underline{n}'^*$ , we have  $\underline{r}' \overline{\times} \underline{n}' = 0$ . Since the direction of the line is independent of the translation of the rigid body motion, the constraint on the non-null part can be used to generate equations with the parameters of the rotation as the only unknowns. The constraint on the null part can then be used to determine the unknown translation. In other words, since the motor to be estimated,  $\underline{M} = \underline{R}'_1 + \underline{e}\underline{R}'_2$ , is determined in its non-null part only by rotation, the non-null part of the constraint allows

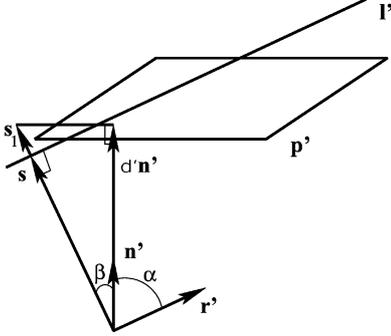


Figure 3. The Euclidean plane  $p'$  is represented by its normal  $n'$  (as bivector) and the Hesse distance  $d'$ . Furthermore, we choose  $s \in l'$  with  $s \perp r'$ . The angle of  $r'$  and  $n'^*$  is  $\alpha$  and the angle of  $s$  and  $n'^*$  is  $\beta$ . We choose the vector  $s_1$  with  $s \parallel s_1$  so that  $d'n'^*$  is the orthogonal projection of  $(s + s_1)$  onto  $n'^*$ .

to estimate the rotor  $R'_1$ , while the null part of the constraint allows to estimate the rotor  $R'_2$ . So it is possible to sequentially separate equations of the unknown rotation from equations of the unknown translation without the limitations known from the embedding of the problem in Euclidean space [6]. This is useful since the two smaller systems of equations are faster to solve than one larger system of equations. To analyze the null part of the constraint we interpret the moment  $m'$  of the line representation  $\underline{L}' = Er' + em'$  as  $m' = s \times r'$  by choosing a vector  $s$  with  $s \in l'$  and  $s \perp r'$ . Following [28], we can calculate

$$\begin{aligned} n' \times m' &= -(s \times r') \times n' \\ &= (s \bar{\times} n') \bar{\times} r' - s \bar{\times} (r' \bar{\times} n'). \end{aligned} \quad (2.11)$$

Now we can evaluate

$$\begin{aligned} dI_{Er'} - (n' \times m') &= dI_{Er'} - (s \bar{\times} n') \bar{\times} r' \\ &\quad + s \bar{\times} (r' \bar{\times} n'). \end{aligned} \quad (2.12)$$

Figure 3 shows the scenario. Further, we can find a vector  $s_1 \parallel s$  with  $0 = d' - (\|s\| + \|s_1\|) \cos(\beta)$ . The vector  $s_1$  might also be anti-parallel to  $s$ . This leads to a change of the sign, but does not affect the constraint itself. Now we can evaluate

$$\begin{aligned} d'I_{Er'} - (n' \times m') &= d'I_{Er'} - \|s\| \cos(\beta) r' + \cos(\alpha) s \\ &= \|s_1\| \cos(\beta) r' + \cos(\alpha) s. \end{aligned} \quad (2.13)$$

Thus, the error of the null part of the motor is constituted by the sum of the vector  $s$ , scaled by the angle  $\alpha$ , and the direction vector  $r'$ , scaled by the norm of  $s_1$  and the angle  $\beta$ .

If  $r' \perp n'^*$ , then  $n'^* \parallel s$  and, thus, we will find

$$\begin{aligned} \|d'I_{Er'} - (n' \times m')\| &= \|d'I_{Er'} + s \bar{\times} (r' \bar{\times} n') \\ &\quad - (s \bar{\times} n') \bar{\times} r'\| \\ &= \|d'I_E \bar{\times} r' - (s \bar{\times} n') \bar{\times} r'\| \\ &= \|d'I_E - (s \bar{\times} n')\|. \end{aligned} \quad (2.14)$$

This means, in agreement with the point-plane constraint, that the above difference measure corresponds the Hesse distance of the line to the plane. Since Eq. (2.13) contains the error vector  $s$ , its error value is dependent on the chosen origin of the vector space. This effect is indeed unwanted and can lead to bad conditioned equations, but if  $n'^*$  is nearly parallel to  $s$  good conditioned equations can be assured since they are then related to the point-plane constraint.

This analysis shows that the considered constraints are not only qualitative constraints, but also quantitative ones. This is very important, since we want to measure the extend of fulfillment of these constraints in the case of noisy data.

#### 2.4. Constraint Equations for Pose Estimation

Now it is possible to express the 2D-3D pose estimation problem in a quantitative manner. The aim is to express that *a rigidly transformed object entity has to lie on a projectively reconstructed image entity* in the conformal geometric algebra. Let  $\underline{X} \in \mathcal{G}_{4,1}$  be an object point and  $\underline{L} \in \mathcal{G}_{4,1}$  be an object line. The (unknown) transformed entities can be written as  $\underline{X}' = M\underline{X}\tilde{M}$  and  $\underline{L}' = M\underline{L}\tilde{M}$ . Let  $x \in \mathcal{G}_{2,1}$  be an image point and  $l \in \mathcal{G}_{2,1}$  be an image line. Note, that we denote the 2D projective image features also with small bold letters, similar to 3D Euclidean points. The reason is, that both algebras are built from three basis vectors, they can not be confounded in the scenario and it avoids extra funds. The projective reconstruction of these entities can be written as  $L_x = O \wedge x \in \mathcal{G}_{3,1}$  and  $P_l = O \wedge l \in \mathcal{G}_{3,1}$ . The point  $O \in \mathcal{G}_{3,1}$  denotes the optical center of the camera. Then we can apply the  $e \wedge$ -operator to change the representations from the projective to the conformal space, and combine it with the commutator and anti-commutator products to express the collinearity and coplanarity of the involved entities.

Thus, the constraint equations of pose estimation read as follows, a) Point-line constraint:

$$\lambda \underbrace{\left( \underbrace{(\mathbf{M} \ \underline{\mathbf{X}} \ \tilde{\mathbf{M}})}_{\text{object point}} \right)}_{\text{rigid motion of the object point}} \times \underbrace{\left( \underbrace{\mathbf{e} \wedge (\mathbf{O} \wedge \mathbf{x})}_{\text{projection ray, reconstructed from the image point}} \right)}_{\text{collinearity of the transformed object point with the reconstructed line}} \cdot \mathbf{e}_+ = 0. \quad (2.15)$$

b) Point-plane constraint:

$$\lambda \underbrace{\left( \underbrace{(\mathbf{M} \ \underline{\mathbf{X}} \ \tilde{\mathbf{M}})}_{\text{object point}} \right)}_{\text{rigid motion of the object point}} \times \underbrace{\left( \underbrace{\mathbf{e} \wedge (\mathbf{O} \wedge \mathbf{I})}_{\text{3D plane, reconstructed from the image line}} \right)}_{\text{coplanarity of the transformed object point with the reconstructed plane}} \cdot \mathbf{e}_+ = 0. \quad (2.16)$$

c) Line-plane constraint:

$$\lambda \underbrace{\left( \underbrace{(\mathbf{M} \ \underline{\mathbf{L}} \ \tilde{\mathbf{M}})}_{\text{object line}} \right)}_{\text{rigid motion of the object line}} \bar{\times} \underbrace{\left( \underbrace{\mathbf{e} \wedge (\mathbf{O} \wedge \mathbf{I})}_{\text{3D plane, reconstructed from the image line}} \right)}_{\text{coplanarity of the transformed object line with the reconstructed plane}} = 0. \quad (2.17)$$

Note, that there is no  $\cdot \mathbf{e}_+$  operation involved in the line-plane constraint. The reason is, that the constraint equation is partitioned into one equation on the non-null part and one equation on the null part of the constraint equation as explained in the last section.

The involved mathematical spaces are exemplarily shown for the point-line constraint,

$$\lambda \underbrace{\left( \underbrace{(\mathbf{M} \ \underline{\mathbf{X}} \ \tilde{\mathbf{M}})}_{\text{CS}} \right)}_{\text{CS}} \times \underbrace{\left( \underbrace{\mathbf{e} \wedge (\mathbf{O} \wedge \mathbf{x})}_{\text{PS PP}} \right)}_{\text{PS}} \cdot \mathbf{e}_+ = 0. \quad (2.18)$$

$\underbrace{\hspace{10em}}_{\text{ES}}$

Here does *PP* abbreviate *projective plane*, *PS* *projective space*, *CS* *conformal space* and *ES* the *Euclidean space*. These compact equations subsume the pose estimation problem at hand: find the best motor  $\mathbf{M}$  which

satisfies the constraint. The 2D-3D pose estimation problem is described in an implicit way. Note, that the stratification hierarchy of the involved entities is strictly kept within these equations. Furthermore are the equations compact and therefore easy to interpret. Additionally, the geometric analysis of the constraints assure well conditioned equations and help to interpret effects of the constraints discussed in the experimental part. The constraints behave robust in case of noisy data, and linearization and iteration enables the design of fast (real-time capable) algorithms. In contrast to other approaches, where the minimization of errors has to be computed directly on the manifold of geometric transformations [5, 41], in our approach a distance in the Euclidean space constitutes the error measure.

This is the now the complete formulation and analysis of the constraint equations already shown in Part I, Section 4.

### 3. Pose Estimation with Extended Object Concepts

This section concerns the development of constraint equations to relate 3D kinematic chains, circles and spheres with corresponding extracted 2D image features. Similar to the previous section we will formalize constraint equations in the 3D space, which contain a geometric distance measure.

#### 3.1. Pose Estimation of Kinematic Chains

So far we have parameterized the 3D pose constraint equations of a rigid object. Let be given a rigid object by a set of entities as points and lines. Assume that a second rigid object is attached to the first one by a joint. The joint can be formalized as an axis of rotation and/or translation in the object frame. If the joint  $j$  is only dependent on a variable angle  $\theta_j$ , it is called a revolute joint, and it is called a prismatic joint if the degree of freedom is only a variable length  $d_j$ . This parameterization of joints is also called the Denavit-Hartenberg parameterization [7]. Each joint defines a new coordinate system, and the coordinate transformations between joints can be expressed by suitable motors  $\mathbf{M}_j$ . This means, an entity given in the coordinate system of the  $j$ -th joint can be translated in an entity of the base coordinate system by transforming it with the motors  $\mathbf{M}_0, \dots, \mathbf{M}_j$ .

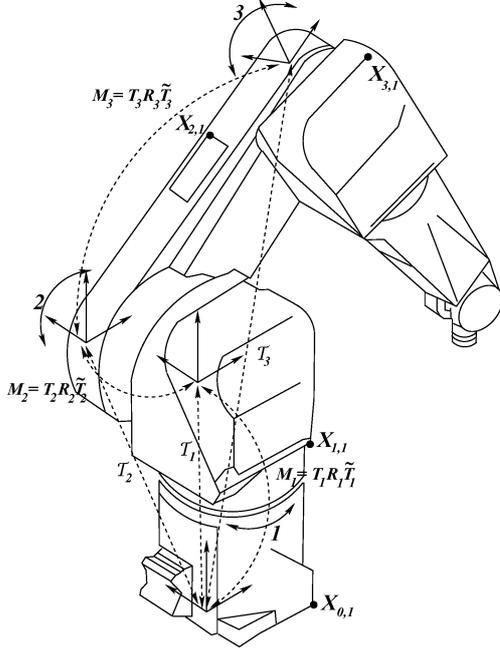


Figure 4. The RX-90 robot arm. The internal joint transformations  $M_j$  and the global transformations  $T_j$  are visualized.

Such objects are also called *kinematic chains*. With kinematic chains we mean flexible linked rigid objects which can only change their pose in mutual dependence. Examples are robot arms [40] or human body movements, see e.g. Fig. 4. Kinematic chains can be parameterized by their including joints. Every joint defines a new coordinate system. To estimate the position of an end-effector entity of a kinematic chain in terms of an other base coordinate system, all involved joint coordinate systems must be traced. This is visualized in Fig. 4. For short notations of the single transformations in CGA between the joints we define

$$\begin{aligned} \mathcal{T}_0\{\underline{\mathbf{X}}_{0,i_0}\} &:= \mathbf{M}_0 \underline{\mathbf{X}}_{0,i_0} \tilde{\mathbf{M}}_0 = \underline{\mathbf{X}}_{0,i_0} \\ \mathcal{T}_j\{\underline{\mathbf{X}}_{j,i_j}, \mathbf{M}_j\} &:= \mathcal{T}_{j-1}\{\mathbf{M}_j \underline{\mathbf{X}}_{j,i_j} \tilde{\mathbf{M}}_j, \mathbf{M}_{j-1}\} : j = 1, \dots, n \\ &= \mathbf{M}_1 \dots \mathbf{M}_j \underline{\mathbf{X}}_{j,i_j} \tilde{\mathbf{M}}_j \dots \tilde{\mathbf{M}}_1 : j = 1, \dots, n. \end{aligned} \quad (3.1)$$

The function  $\mathcal{T}_0$  with the motor  $\mathbf{M}_0$  describes the identity for points which are not subject to internal transformations. We call them *base points*. Since the motor  $\mathbf{M}_0$  is just the identity it will be neglected in the further equations. The function  $\mathcal{T}_j$  formalizes the transformation of an attached joint  $j$  with respect to the basis coordinate system in an inductive manner. In the general

case, the transformation of a point  $\underline{\mathbf{X}}_{j,i_j}$  of a  $j$ -th joint to the base coordinate system is represented by a sequence of such motors  $\mathbf{M}_1, \dots, \mathbf{M}_j$ . An object model  $\mathcal{O}$  of a kinematic chain with  $n$  segments can now be represented by a set of  $n + 1$  such functions  $\mathcal{T}_j$ ,

$$\begin{aligned} \mathcal{O} &= \{\mathcal{T}_0\{\underline{\mathbf{X}}_{0,i_0}\}, \mathcal{T}_1\{\underline{\mathbf{X}}_{1,i_1}, \mathbf{M}_1\}, \dots, \\ &\quad \mathcal{T}_n\{\underline{\mathbf{X}}_{n,i_n}, \mathbf{M}_n\} \mid n, i_0, \dots, i_n \in \mathbb{N}\}. \end{aligned}$$

### 3.2. Constraint Equations of Kinematic Chains

Now we will combine the introduced representation of kinematic chains in CGA with the pose estimation constraints derived in Section 2. This is very simple now because everything is formulated in the same algebra. Note, that the constraints are presented unscaled, so the  $\lambda(\cdot) \cdot \mathbf{e}_+$  operation is not extra written.

The general unknown pose corresponds to a motor  $\mathbf{M}$ . For the base points  $\underline{\mathbf{X}}_{0,i_0}$  the constraint equations reduce for a suitable projection ray  $\underline{\mathbf{L}}_{0,i_0} = \mathbf{e} \wedge (\mathbf{O} \wedge \mathbf{x}_{0,i_0})$  to

$$\begin{aligned} (\mathbf{M}(\mathcal{T}_0\{\underline{\mathbf{X}}_{0,i_0}\})\tilde{\mathbf{M}}) \underline{\mathbf{x}} \wedge (\mathbf{O} \wedge \mathbf{x}_{0,i_0}) &= 0 \\ \Leftrightarrow (\mathbf{M}\underline{\mathbf{X}}_{0,i_0}\tilde{\mathbf{M}}) \underline{\mathbf{x}} \wedge (\mathbf{O} \wedge \mathbf{x}_{0,i_0}) &= 0. \end{aligned} \quad (3.2)$$

The general constraint equation for a point  $\underline{\mathbf{X}}_{j,i_j}$  at the  $j$ -th joint leads to

$$\begin{aligned} (\mathbf{M}(\mathcal{T}_j\{\underline{\mathbf{X}}_{j,i_j}, \mathbf{M}_j\})\tilde{\mathbf{M}}) \underline{\mathbf{x}} \wedge (\mathbf{O} \wedge \mathbf{x}_{j,i_j}) &= 0 \\ \Leftrightarrow (\mathbf{M}(\mathbf{M}_1 \dots \mathbf{M}_j \underline{\mathbf{X}}_{j,i_j} \tilde{\mathbf{M}}_j \dots \tilde{\mathbf{M}}_1)\tilde{\mathbf{M}}) \underline{\mathbf{x}} \wedge & \\ \wedge (\mathbf{O} \wedge \mathbf{x}_{j,i_j}) &= 0. \end{aligned} \quad (3.3)$$

It is also simple to use extracted image lines  $\underline{\mathbf{l}}_{j,i_j}$  and their reconstructed projection planes  $\underline{\mathbf{p}}_{j,i_j} = \mathbf{e} \wedge (\mathbf{O} \wedge \underline{\mathbf{l}}_{j,i_j})$ . For such situations, the constraint equations reduce to

$$\begin{aligned} (\mathbf{M}(\mathcal{T}_0\{\underline{\mathbf{X}}_{0,i_0}\})\tilde{\mathbf{M}}) \underline{\mathbf{x}} \wedge (\mathbf{O} \wedge \underline{\mathbf{l}}_{0,i_0}) &= 0 \\ \Leftrightarrow (\mathbf{M}\underline{\mathbf{X}}_{0,i_0}\tilde{\mathbf{M}}) \underline{\mathbf{x}} \wedge (\mathbf{O} \wedge \underline{\mathbf{l}}_{0,i_0}) &= 0 \end{aligned} \quad (3.4)$$

for the base points, and the general constraint equation for a point at the  $j$ th joint leads to

$$\begin{aligned} (\mathbf{M}(\mathcal{T}_j\{\underline{\mathbf{X}}_{j,i_j}, \mathbf{M}_j\})\tilde{\mathbf{M}}) \underline{\mathbf{x}} \wedge (\mathbf{O} \wedge \underline{\mathbf{l}}_{j,i_j}) &= 0 \\ \Leftrightarrow (\mathbf{M}(\mathbf{M}_1 \dots \mathbf{M}_j \underline{\mathbf{X}}_{j,i_j} \tilde{\mathbf{M}}_j \dots \tilde{\mathbf{M}}_1)\tilde{\mathbf{M}}) \underline{\mathbf{x}} \wedge & \\ \wedge (\mathbf{O} \wedge \underline{\mathbf{l}}_{j,i_j}) &= 0. \end{aligned} \quad (3.5)$$

We can also describe kinematic chains by lines and combine them with the line-plane-constraint. For this, only lines  $\underline{L}_{j,i_j}$  and projection planes  $\underline{P}_{j,i_j} = \mathbf{e} \wedge (\mathbf{O} \wedge \mathbf{l}_{j,i_j})$  have to be substituted and combined with the anticommutator product. For the base lines we get

$$\begin{aligned} (\mathbf{M}(\mathcal{T}_0\{\underline{L}_{0,i_0}\})\tilde{\mathbf{M}}) \bar{\times} \mathbf{e} \wedge (\mathbf{O} \wedge \mathbf{l}_{0,i_0}) &= 0 \\ \Leftrightarrow (\mathbf{M}\underline{L}_{0,i_0}\tilde{\mathbf{M}}) \bar{\times} \mathbf{e} \wedge (\mathbf{O} \wedge \mathbf{l}_{0,i_0}) &= 0, \end{aligned} \quad (3.6)$$

and for a line on the  $j$ -th joint we get

$$\begin{aligned} (\mathbf{M}(\mathcal{T}_j\{\underline{L}_{j,i_j}, \mathbf{M}_j\})\tilde{\mathbf{M}}) \bar{\times} \mathbf{e} \wedge (\mathbf{O} \wedge \mathbf{l}_{j,i_j}) &= 0 \\ \Leftrightarrow (\mathbf{M}(\mathbf{M}_1 \dots \mathbf{M}_j \underline{L}_{j,i_j} \tilde{\mathbf{M}}_j \dots \tilde{\mathbf{M}}_1) \tilde{\mathbf{M}}) \\ \bar{\times} \mathbf{e} \wedge (\mathbf{O} \wedge \mathbf{l}_{j,i_j}) &= 0. \end{aligned} \quad (3.7)$$

A description of kinematic chains in CGA and the construction of pose estimation constraints can also be found in [30].

### 3.3. Pose Estimation Using Constraints for Circles and Spheres

In this section constraint equations are derived to relate 3D circles to 2D conics and 3D spheres to 2D conics. We start with an analysis of involved problems and then present a suitable solution approach.

#### 3.3.1. The Problem of Tangentiality Constraints.

Since also the constraints for circles and spheres have to be derived in the 3D space, the aim is to reconstruct certain entities from image information and to compare the reconstructed entities with the 3D model entities. The reconstruction based on an image conic (the image of a circle or sphere) leads to a cone. Indeed, we can not formalize cones as single entities in conformal geometric algebra. But to enable the above mentioned comparison, we formalize constraint equations for tangentiality of 3D circles or spheres to projection rays, reconstructed from image points of the corresponding image entity. We denote the spatial tangentiality of a 3D circle  $\underline{Z}$  to a 3D line  $\underline{L}$  as circle-line constraint and the tangentiality of a 3D sphere  $\underline{S}$  to a 3D line  $\underline{L}$  as sphere-line constraint. Figure 5 visualizes the idea.

It is very easy in CGA to express e.g. tangentiality of a non-coplanar line  $\underline{L}$  to a circle  $\underline{Z}$ : The point  $\underline{X}_z := \underline{L} \vee \underline{Z}$  is a null vector (this means  $\underline{X}_z^2 = 0$ ) iff the entities intersect. But this only holds in ideal geometry. In reality, there are several cases how a line can be related to a circle: it can intersect, be coplanar or

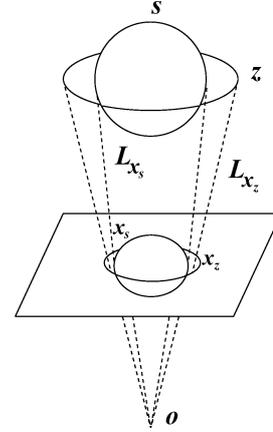


Figure 5. Visualization of the circle-line and sphere-line constraint in conformal geometric algebra.

perpendicular. The line can pass *outside* or *inside* the circle, etc. By defining a line in a parameterized manner, it is easy to see that the error function of points on a line to a circle can contain one global minimum, two global minima, one local and one global minima or no minimum in non-degenerate and degenerate cases. In Fig. 6 three example lines are shown: Two lines are parallel to the plane, in which the circle lies. One of these lines passes the circle *outside*, the other one *inside*. This leads to error functions, containing one global minimum or two global minima. The third line is passing the *inside* of the circle and is not parallel to the plane in which the circle lies. This results in one global and one local minimum. From that relations result two possible strategies: First, we make a case decision, depending on the geometric situation. This is hard to implement and to combine with our previous derived constraint equations. Second, we can parameterize the circle in a suitable way. This will be done in the following section.

Comparing spheres with lines is in ideal geometry also no problem. One short way to formalize tangentiality is to estimate the distance from the center of the sphere to the line and to subtract the radius: Let  $\underline{L}'$  be the scaled line, as described in Section 2. The line  $\underline{L}'$  is tangential to  $\underline{S} = \underline{P} - \rho^2 \mathbf{e}$  iff

$$\|(\underline{L}' \times \underline{P}) \cdot \mathbf{e}_+\| - \rho = 0. \quad (3.8)$$

The main problem in this formulation is the square root term of the norm containing the unknowns in quadratic terms since  $\|\mathbf{x}\| = \sqrt{\sum(x_i)^2}$ . This leads to equations, which are not nice to handle if we want to estimate the

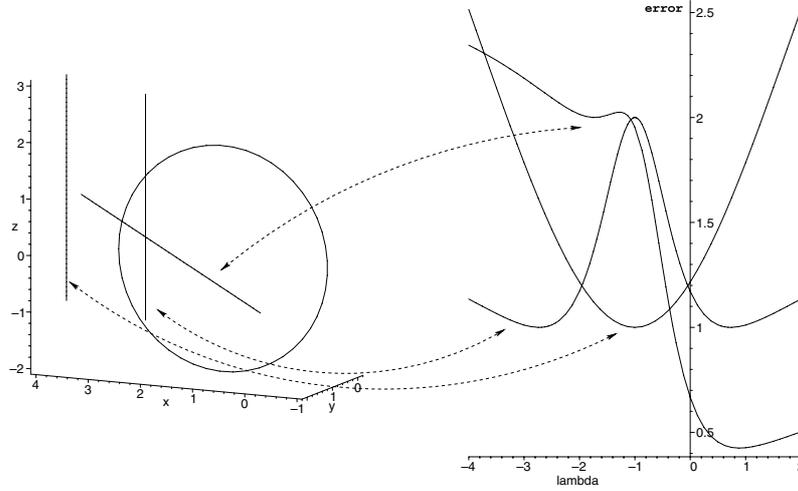


Figure 6. Different geometric relations of lines to circles leads to different kinds of error functions for parameterized lines.

unknown motor  $M$  in the equation

$$\sqrt{((M\underline{P}\tilde{M} \times \underline{L}') \cdot \mathbf{e}_+)^2} - \rho = 0. \quad (3.9)$$

We made experiments with these kind of equations and implemented a Newton-Raphson method to solve the equations. But there are two main problems: First, the convergence rate is very slow and the algorithm often converges against the wrong minimum (the algorithm needs about 5 seconds to estimate the pose). Second, we loose the possibility to combine them with the other constraints for simultaneous considerations in pose estimation.

The key idea to relate circles and spheres to lines, is to interpret the circles and spheres as orbits generated by twist operations by modeling general rotations as introduced in the next section.

**3.3.2. Operational Definition of Circles and Spheres Using Twists.** We will first repeat the general description of circles and spheres, as introduced in Part I and then generate an operational definition of these entities. In the next section we will continue with the formulations of the circle-line and sphere-line constraints.

Let be  $\underline{Z} = \underline{A} \wedge \underline{B} \wedge \underline{C}$  a circle in CGA. Evaluating the outer products of three points leads to

$$\underline{Z} = \underline{A} \wedge \underline{B} \wedge \underline{C} = A + A^- \mathbf{e} + A^+ \mathbf{e}_0 + A^\pm \mathbf{E} \quad (3.10)$$

with suitable multivectors  $A$ ,  $A^-$ ,  $A^+$  and  $A^\pm$ , see Part I.

A circle can also be understood as a twist  $\underline{L}_z$  modeling a general rotation and a point  $\underline{X}_z$  on the circle. From the dual representation of the circle, this information is very easy to extract since the generating twist parameters are directly given. The twist transformation corresponds to a suitable parameterized motor  $M_\phi$ ,

$$M_\phi = \exp\left(\frac{\phi}{2}(A^+ + \mathbf{e}A^\pm)\right). \quad (3.11)$$

The points on the circle are simply given by

$$\underline{X}_z^\phi = (M_\phi \underline{X}_z \tilde{M}_\phi) : \phi \in [0, \dots, 2\pi]. \quad (3.12)$$

Figure 7 (left) visualizes the geometry. The circle results as the orbit of the unique motor, which moves a certain point.

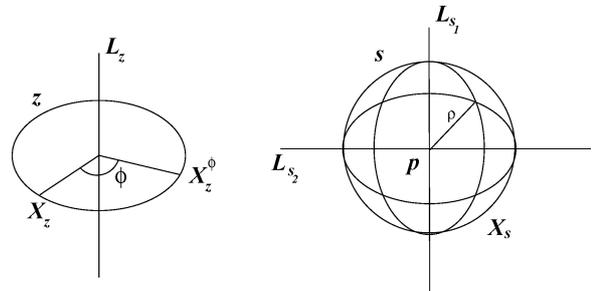


Figure 7. Circles and spheres parameterized with twists.

Now we continue with the formalization of a sphere. The general expression of a sphere leads to

$$\underline{S} = \left( \underline{P} - \frac{1}{2}\rho^2\mathbf{e} \right). \quad (3.13)$$

In this formulation,  $\underline{P}$  is the center of the sphere and  $\rho$  is the radius.

The idea is to formalize spheres in an operational manner as two coupled twists modeling general rotations. In that approach a sphere is formalized by a point  $\underline{X}_s$  on a sphere and two perpendicular twists,  $\underline{L}_{s_1}$  and  $\underline{L}_{s_2}$ , intersecting in the origin of the sphere. Figure 7 (right) visualizes the idea. The corresponding motors are denoted as  $M_{\phi_1}$  and  $M_{\phi_2}$ ,

$$\begin{aligned} M_{\phi_1} &= \exp\left(\frac{\phi_1}{2}(\mathbf{e}_{12} + \mathbf{e}(\mathbf{p} \cdot \mathbf{e}_{12}))\right), \\ M_{\phi_2} &= \exp\left(\frac{\phi_2}{2}(\mathbf{e}_{31} + \mathbf{e}(\mathbf{p} \cdot \mathbf{e}_{31}))\right). \end{aligned} \quad (3.14)$$

The bivectors  $\mathbf{e}_{12}$  and  $\mathbf{e}_{31}$  are the two perpendicular rotation planes, belonging to the rotation axes which are connected to the center  $\underline{P}$  of the sphere. Then all points on the sphere  $\underline{S}$  result from the equation

$$\underline{X}_s^{\phi_1, \phi_2} = (M_{\phi_1} M_{\phi_2} \underline{X}_s \tilde{M}_{\phi_2} \tilde{M}_{\phi_1}) : \phi_1, \phi_2 \in [0, \dots, 2\pi]. \quad (3.15)$$

This principle of coupling two motors is virtual in contrast to kinematic chains, which correspond the coupling of physical objects. The principle of virtual coupling can be further extended to construct more complex orbits of twists and, thus, to enable pose estimation of more complex objects, see [33].

**3.3.3. The Constraint Equations of Circles and Spheres to Lines.** So far we have developed the formalization of circles and spheres as orbits of twists. We will use these representations to express incidence of circles  $\underline{Z}$  and spheres  $\underline{S}$  to 3D lines  $\underline{L}$ .

While in the constraint equations of Section 2 the motors are the only unknowns to be estimated, now we have higher loads because of the parameterization of the features or entities of pose estimation.

We will start with the formalization of a suitable circle-line constraint. To relate the circle  $\underline{Z}$  to a line  $\underline{L} = \mathbf{e} \wedge (\mathbf{O} \wedge \mathbf{x})$ , we only need to estimate the unknown angle  $\phi$ , which leads to collinearity of the suitable transformed point  $\underline{X}_z \in \underline{Z}$  to  $\underline{L}$ . The circle-line

constraint can now be written as

$$(M_{\phi} \underline{X}_z \tilde{M}_{\phi}) \underline{\times} \mathbf{e} \wedge (\mathbf{O} \wedge \mathbf{x}) = 0. \quad (3.16)$$

In this equation, the angle  $\phi$  is an additional unknown for each constraint equation. The pose estimation constraint equation for an unknown rigid body motion now means to estimate both the best motor  $M$  and the angle  $\phi$ ,

$$(M(M_{\phi} \underline{X}_z \tilde{M}_{\phi}) \tilde{M}) \underline{\times} \mathbf{e} \wedge (\mathbf{O} \wedge \mathbf{x}) = 0. \quad (3.17)$$

The sphere-line constraint, respectively the incidence of a line  $\underline{L} = \mathbf{e} \wedge (\mathbf{O} \wedge \mathbf{x})$  to a sphere  $\underline{S}$  can be described by a point  $\underline{X}_s$  on the sphere and the two motors  $M_{\phi_1}$  and  $M_{\phi_2}$ ,

$$(M_{\phi_1} M_{\phi_2} \underline{X}_s \tilde{M}_{\phi_2} \tilde{M}_{\phi_1}) \underline{\times} \mathbf{e} \wedge (\mathbf{O} \wedge \mathbf{x}) = 0. \quad (3.18)$$

In this constraint equation  $\phi_1$  and  $\phi_2$  are additional unknowns. The pose estimation constraint equation for an unknown rigid body motion means to estimate the best motor  $M$  and the two angles  $\phi_1$  and  $\phi_2$ ,

$$(M(M_{\phi_1} M_{\phi_2} \underline{X}_s \tilde{M}_{\phi_2} \tilde{M}_{\phi_1}) \tilde{M}) \underline{\times} \mathbf{e} \wedge (\mathbf{O} \wedge \mathbf{x}) = 0. \quad (3.19)$$

This approach to formalize constraint equations for circles and spheres appears surprising in the context of our algebraic embedding. The main problem with these entities is, how to formalize constraint equations, which obtain the characteristics mentioned in Part I. For this reason we choose an operational definition of circles and spheres and linearize them in the same manner as we linearize the pose problem: We formulate these entities in their tangential space and choose a Lie algebra representation of these entities.

## 4. Real-Time Pose Estimation

This section concerns the numerical estimation of the pose parameters and presents experimental results.

### 4.1. Estimation of Motion Parameters

In the last sections, several constraint equations to relate object information to image information are derived. In these equations, the object, camera and image information is assumed to be known and the motor  $M$  expressing the motion is assumed to be unknown. The main question is now, how to solve a set of constraint equations for multiple (different) features with respect

to the unknown motor  $\mathbf{M}$ . Since a motor is a polynomial of infinite degree (see, e.g., its series expression), this is a non-trivial task, especially in the case of real-time estimations.

**4.1.1. Linearization in the Tangential Space.** The idea is to gain linear equations with respect to the generators of the motor. We use the exponential representation of motors and apply the Taylor series expression of first order for approximation. This leads to a mapping of the above mentioned global motion transformation to a twist representation, which enables incremental changes of pose. That means, we do not search for the parameters of the Lie group  $SE(3)$  to describe the rigid body motion [11], but for the parameters which generate their Lie algebra  $se(3)$  [24]. This results in linear equations in the generators of the unknown 3D rigid body motion. In this section we derive the linearization of the motors. For the sake of simplicity we will do that in the case of point transformations. The Euclidean transformation of a point  $\underline{\mathbf{X}}$  caused by the motor  $\mathbf{M}$  is approximated in the following way:

$$\begin{aligned} \mathbf{M}\underline{\mathbf{X}}\tilde{\mathbf{M}} &= \exp\left(-\frac{\theta}{2}(\mathbf{l}' + \mathbf{e}\mathbf{m}')\right)\underline{\mathbf{X}}\exp\left(\frac{\theta}{2}(\mathbf{l}' + \mathbf{e}\mathbf{m}')\right) \\ &\approx \left(1 - \frac{\theta}{2}(\mathbf{l}' + \mathbf{e}\mathbf{m}')\right)\underline{\mathbf{X}}\left(1 + \frac{\theta}{2}(\mathbf{l}' + \mathbf{e}\mathbf{m}')\right) \\ &\approx \mathbf{E} + \mathbf{e}(\mathbf{x} - \theta(\mathbf{l}' \cdot \mathbf{x}) - \theta\mathbf{m}'). \end{aligned} \quad (4.1)$$

Setting  $\mathbf{l} := \theta\mathbf{l}'$  and  $\mathbf{m} := \theta\mathbf{m}'$  leads to

$$\mathbf{M}\underline{\mathbf{X}}\tilde{\mathbf{M}} \approx \mathbf{E} + \mathbf{e}(\mathbf{x} - \mathbf{l} \cdot \mathbf{x} - \mathbf{m}). \quad (4.2)$$

By combining this approximation of the motion with the previously derived constraints (e.g. the point-line constraint) this leads to<sup>1</sup>

$$\begin{aligned} 0 &= \mathbf{M}\underline{\mathbf{X}}\tilde{\mathbf{M}} \times \underline{\mathbf{L}} \\ \Leftrightarrow 0 &= \exp\left(-\frac{\theta}{2}(\mathbf{l}' + \mathbf{e}\mathbf{m}')\right)\underline{\mathbf{X}} \\ &\quad \times \exp\left(\frac{\theta}{2}(\mathbf{l}' + \mathbf{e}\mathbf{m}')\right) \times \underline{\mathbf{L}} \\ \Leftrightarrow \approx \Rightarrow 0 &= (\mathbf{E} + \mathbf{e}(\mathbf{x} - \mathbf{l} \cdot \mathbf{x} - \mathbf{m})) \times \underline{\mathbf{L}} \\ \Leftrightarrow 0 &= \lambda(\mathbf{E} + \mathbf{e}(\mathbf{x} - \mathbf{l} \cdot \mathbf{x} - \mathbf{m})) \times \underline{\mathbf{L}}. \end{aligned} \quad (4.3)$$

Because of the approximation ( $\Leftrightarrow \approx \Rightarrow$ ) the unknown motion parameters  $\mathbf{l}$  and  $\mathbf{m}$  are linear. This equation contains six unknown parameters for the rigid body motion. The unknowns are the six unknown twist parameters for the screw motion. In the last step we scale the linearized constraints with a suitable factor  $\lambda$  to

express an Euclidean distance measure as explained in Section 2. This means, everything so far happens unscaled and only in the very last step we scale the constraint equation and go to the Euclidean space, as one of the strata of the hierarchy described in Part I.

The linear equations can be solved for a set of correspondences by applying e.g. the Householder method [29]. From the solution of the system of equations, the motion parameters  $\mathbf{R}$ ,  $\mathbf{t}$  can easily be recovered by evaluating  $\theta := \|\mathbf{l}\|$ ,  $\mathbf{l}' := \frac{\mathbf{l}}{\theta}$  and  $\mathbf{m}' := \frac{\mathbf{m}}{\theta}$ . The Motor  $\mathbf{M}$  can be evaluated by applying the Rodrigues' formula [11, 25]. This procedure is iterated to converge to the whole rigid motion. Figure 8 visualizes the principle of such an approximation and iteration: The aim is to rotate a point  $\underline{\mathbf{X}}$  around 90 degrees to a point  $\underline{\mathbf{X}}'$ . The first order approximation of the rotation leads to the tangent of the circle passing through  $\underline{\mathbf{X}}$ . Normalizing the tangent line to  $\underline{\mathbf{X}}'$  (denoted by dashed lines) we get  $\underline{\mathbf{X}}_1$  as the first order approximation of the required point  $\underline{\mathbf{X}}'$ . By repeating this procedure the points  $\underline{\mathbf{X}}_2, \dots, \underline{\mathbf{X}}_n$  will be estimated, which converge to the point  $\underline{\mathbf{X}}'$ . It is clear from Fig. 8 that the convergence rate of a rotation is dependent on the amount of the expected rotation. An analysis of the convergence rate for general angles is given in the next section.

Note, that basically this estimation procedure corresponds to a gradient descent method in the 3D space.

#### 4.1.2. Generating an Example System of Equations.

In this section we will derive a system of equations for point, line and plane correspondences to visualize the type of equations which are obtained.

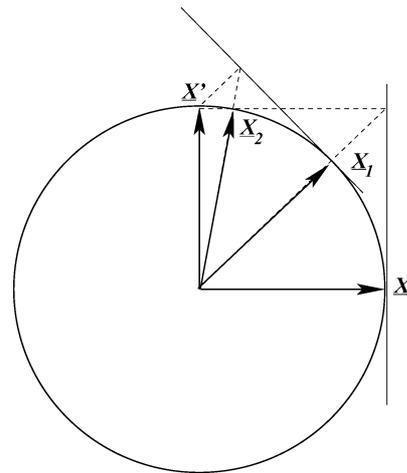


Figure 8. Principle of the convergence rate for the iteration of a point  $\underline{\mathbf{X}}$  rotated around 90 degrees to a point  $\underline{\mathbf{X}}'$ .  $\underline{\mathbf{X}}_1$  is the result of the first iteration and  $\underline{\mathbf{X}}_2$  is the result of the second iteration.

Let us assume two points

$$\mathbf{P}_1 = (p_{11}, p_{12}, p_{13}) \quad (4.4)$$

$$\mathbf{P}_2 = (p_{21}, p_{22}, p_{23}), \quad (4.5)$$

one corresponding line (containing a direction  $\mathbf{L}_{d1}$  and a moment  $\mathbf{L}_{m1}$ ),

$$\begin{aligned} \mathbf{L}_1 &= \{\mathbf{L}_{d1} = (L_{d11}, L_{d12}, L_{d13}), \\ &\quad \mathbf{L}_{m1} = (L_{m11}, L_{m12}, L_{m13})\} \end{aligned} \quad (4.6)$$

and one plane  $\mathbf{E}_1$  (containing a normal  $\mathbf{P}_{d1}$  and Hesse distance  $h_{d1}$ ),

$$\mathbf{E}_1 = \{\mathbf{P}_{d1} = (P_{d11}, P_{d12}, P_{d13}), h_{d1}\}. \quad (4.7)$$

Let us further assume that  $\mathbf{P}_1$  corresponds to  $\mathbf{L}_1$  and  $\mathbf{P}_2$  corresponds to  $\mathbf{E}_1$ .

Then the matrix for the system of equations takes the form

$$Ax = b, \quad (4.8)$$

with

$$A = \begin{pmatrix} 0 & L_{d13} & -L_{d12} & -p_{13}L_{d13} - p_{12}L_{d12} & p_{11}L_{d12} & p_{11}L_{d13} \\ -L_{d13} & 0 & L_{d11} & p_{12}L_{d11} & -p_{11}L_{d11} - p_{13}L_{d13} & p_{12}L_{d13} \\ L_{d12} & -L_{d11} & 0 & p_{13}L_{d11} & p_{13}L_{d12} & -p_{12}L_{d12} - p_{11}L_{d11} \\ -P_{d11} & -P_{d12} & -P_{d13} & -P_{d13}p_{22} + P_{d12}p_{23} & P_{d13}p_{21} - P_{d11}p_{23} & -P_{d12}p_{21} + P_{d11}p_{22} \end{pmatrix}.$$

The first three rows contain the components for a point-line constraint and the fourth row the components for a point-plane constraint. The solution vector  $b$  takes the form

$$\begin{aligned} b &= (-p_{12}L_{d13} + p_{13}L_{d12} + L_{m11}, -p_{13}L_{d11} \\ &\quad + p_{11}L_{d13} + L_{m12}, -p_{11}L_{d12} + p_{12}L_{d11} \\ &\quad + L_{m13}, -h_{d1} + P_{d11}p_{21} + P_{d12}p_{22} + P_{d13}p_{23})^T. \end{aligned} \quad (4.9)$$

The system of equations contains as unknowns the six twist parameters for which the equations are solved for. The matrices involving kinematic chains, circles and spheres take a comparable form, just modified with additional unknowns.

Note, that though the point correspondences give three equations the rank is just two. This shows the

well-known fact, that at least three point correspondences are necessary to solve the 2D-3D pose estimation problem. Furthermore gives every point-plane constraint exactly one equation. So at least six correspondences are necessary to get a unique solution.

**4.1.3. Solving the System of Equations.** Many algorithms can be found in the literature to estimate coefficients of non-linear equations systems. A comparison of four approaches for pose estimation are made by Lorusso et al. in [21]. The algorithms deal with 3D point based pose estimation and are based on a SVD decomposition, unit quaternion (UQ), dual quaternion and *eigensystem* (OM) computation. The comparison consists of three parts, *accuracy*, *stability* and *relative efficiency*. Their results are not in agreement with results presented in [42] and they figured out, that the SVD and UQ methods are very similar and usually the most stable. The OM method is not as stable for planar data sets, but superior for large degenerate data sets. The DQ algorithm was never the most stable and usually broke down before the others. Unfortunately they do not compare a gradient descent method within this context. A gradient descent method will be proposed in this work. Therefore we will now study the

convergence rate of the gradient descent method for the case of one unknown angle  $\theta$ . The result is demonstrated in Fig. 9. The  $x$ -axis represents the wanted angle  $\theta$ , the  $y$ -axis shows the estimated angle  $\hat{\theta}$ . Four iterations are overlaid. The functions are very characteristic and it can be seen that the contribution of the first iteration to gain a 90 degree rotation is 45 degree. This becomes clear by comparing the situation with Fig. 8. All angles, except that of 180 degree converge during the iteration, and for the most cases only a few iterations are sufficient to get a good approximation. In situations where only small rotations are assumed, for the most cases, two or three iterations are sufficient.

A comparison of this gradient descent method with a standard SVD-approach or Kalman filter will be done in the first experiment of the next section. There also the adaptive use of pose constraints is presented in more detail.

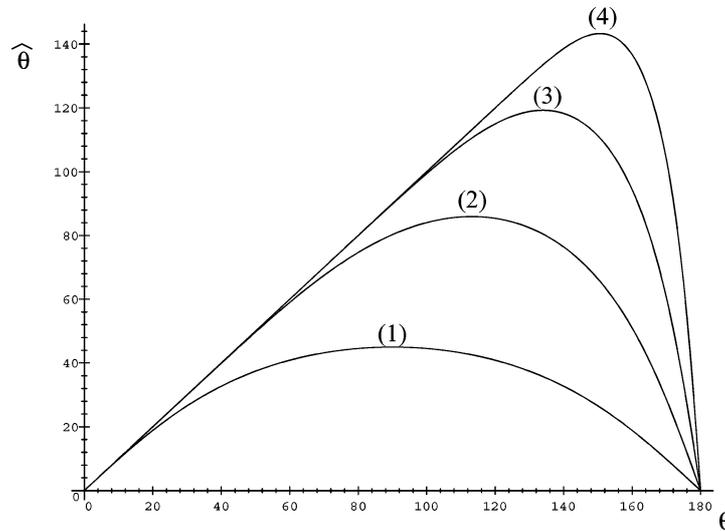


Figure 9. Convergence rate of iterations for arbitrary angles between 0 and 180 degrees. The expected angles  $\theta$  are on the  $x$ -axis and the estimated angles  $\hat{\theta}$  are on the  $y$ -axis. The iterations (1) . . . (4) are overlaid.

#### 4.2. Pose Estimation Experiments

This section shows experimental results which demonstrate that the theoretical approaches for pose estimation developed so far are extremely useful. The first experiment concerns the numerical analysis of the pose estimation algorithm and compares results of the gradient descent method with an SVD-approach and a Kalman filter. The second experiment concerns pose estimation of rigid objects containing points and lines. We show results on real images and explain how to combine the constraints and how to use them in a noise adaptive manner. Then, experiments with kinematic chains are presented. Finally, we describe experiments with *complicated* objects, which contain all different entities we have introduced so far. All information is used to estimate the pose and kinematic chain parameters of the objects simultaneously. The assumptions for our experiments are the following:

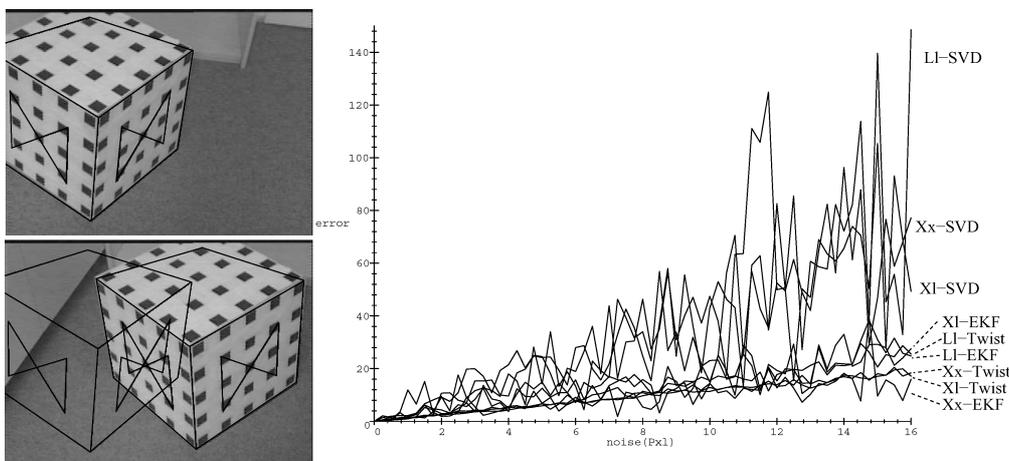
1. Corner features in the image are either manually extracted, or estimated by tracked point markers.
2. Edge features in the image are either reconstructed from two corners or estimated by applying a Hough transformation.
3. Image points on circles or conics are either extracted manually or by a contour algorithm on a silhouette.
4. We use a monocular (calibrated) camera. Only the projection matrix is given, we need no separation of the matrix into intrinsic and extrinsic camera parameters.

5. The 3D (Euclidean) object model is given in terms of feature sets on the object model (corners, lines, kinematic chain locations, etc.)

As explained in the previous section, since we only iterate linear equations containing always six unknowns for the rigid body motion and a few additional ones for the kinematic chains, the pose estimation itself can be carried out in real-time. So far, we are able to estimate the pose of an unknown object by given correspondences and projection matrices in the frame rate of 20 frames per second (fps) on a SUN Ultra 10 and a frame rate of 100 fps on a Linux 2 GHz machine. Note, that the equations are good conditioned with respect to the number of extracted and used image and object features.

##### 4.2.1. Pose Estimation of Simple Rigid Objects.

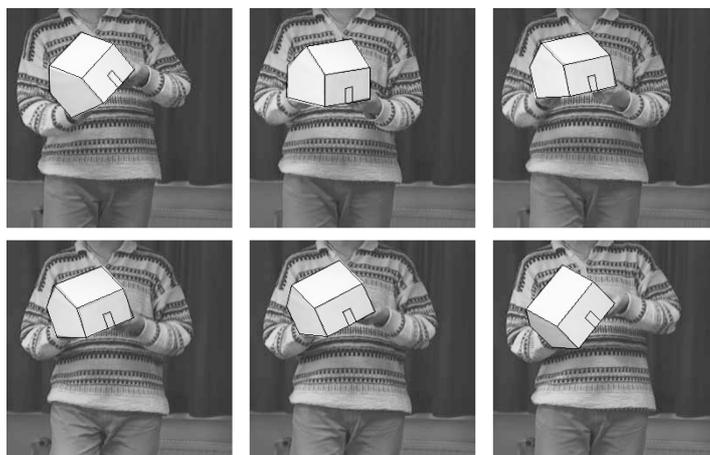
There exist several ways to estimate the motion parameters. In earlier works we concerned this problem and we estimated the motion parameters either on the Lie group  $SE(3)$  itself (by using an SVD approach), or by using an extended Kalman filter (EKF) [39]. In our first experiment, we compare the noise sensitivity of these three methods (the two older ones, and the gradient descent method presented in the last section), with respect to the three constraint equations, relating 3D points to 2D points (Xx), 3D points to 2D lines (Xl), or 3D lines to 2D lines (Ll). Therefore we add a Gaussian noise on extracted image points in a virtual scenario



*Figure 10.* The scenario of the first experiment. In the first image the calibration is performed and the 3D object model is projected on the image. Then the camera moved and corresponding line segments are extracted. For comparison reasons, the initial pose is overlaid. The diagram shows the performance comparison of different methods in case of noisy data.

(see Fig. 10). Then we estimate the rigid body motion, and use the translational error between the ground truth and the disturbed values as error measure. The result is depicted in Fig. 10. It is easy to see, that the results, obtained with the SVD approach are the worst ones. Instead, the Kalman filter and the twist approach have a more stable and comparable error behavior. It is obvious, that the results of the experiments are not much affected by the used constraints themselves. This occurs because we selected certain points directly by hand and derived from these the line subspaces. So the quality of the line subspaces is directly connected to the quality of the point extraction. The result of this investigation is, that for noise corresponding to a distribution

function, the Kalman filter or twist approach for pose estimation should be used. There are two main reasons, why we further prefer the twist approach for pose estimation instead of the EKF: Firstly, the Kalman filter is sensitive to outliers (see e.g. the scenario addressed in Fig. 17), leading to non-converging results. Secondly, Kalman filters must be designed for special situations or scenarios. So the design of a general Kalman filter, dealing with different entities in a weighted manner is hard to implement. Instead, this can be done very easily in the twist approach since the linearized constraint equations of any entity can just be scaled and put in one system of equations. Figure 11 shows results of an automatic tracking algorithm developed



*Figure 11.* Tracking a model house consisting of points and lines.

and analyzed in [31]. The tracking algorithm is a heuristic which relies upon a combination of iterative improvement and random sampling. Iterative improvement refers to a repeated generate-and-test principle by which the algorithm moves from an initial state to its local optimum, see also [2]. We use this approach for self localization and robot navigation tasks.

#### 4.2.2. Adaptive Use of Pose Estimation Constraints.

Image preprocessing algorithms sometimes enable a characterization of the quality of extracted image data (see e.g. [10]). The resulting question is how to deal with noisy extracted image data. The idea to cope with this problem in the context of pose estimation is very simple: Every constraint equation of an image feature describes a distance measure of the involved entity. This constraint equation can be scaled by a factor  $\lambda \in \mathbb{R}$  and so it is possible to individually scale the weights of the equations within the whole system of equations of an observed object. Figure 12 shows an example: We have only three extracted image points and three extracted image lines at hand (see left image). We can use both types of information separately to estimate the pose of the object. Since we have only a few information for each type of correspondences, the object itself is not very well fitted to the image data, see e.g. the upper left or lower right images. On the other hand, we can put both constraint equations in one single system of equations and solve the unknowns by using all available image information simultaneously. Furthermore, we are able to choose different weights of the constraints. The change of the estimated pose is visualized in the other images of Fig. 12. This experiment demonstrates that the presented approach enables to model adaptive observer behavior in a cognitive man-

ner with respect to both the choice of image features at hand and with respect to take into account the trustworthiness of the data.

In an other experiment we simulate the possibility of noise adaptive use of the pose estimation constraints. For this we add a Gaussian noise on some of the extracted image points. Although we know from the problem of Gaussian noise modeling on the unit sphere [5], we will omit these problems here. In this experiment we work with six image features and add on two of them the Gaussian noise. Then we solve the constraint equations with and without weighting the constraints, depending on the noise level. The weights are chosen inverse proportional to the noise level. This means that the more noisy correspondences influence the whole result to a lesser extend. To compare the pose estimation results, we use those without noise as ground truth. We repeat the experiment for every noise level several times to get a smooth error function and choose for every noise level the mean value. The result is visualized in Fig. 13. It is easy to see, that we can use the constraints in a noise adaptive manner.

Figure 14 visualizes the *depth-dependence* of the used 3D constraints. As discussed in Part I, there is a difference of building constraints in the 3D space or in the 2D image plane: The noise in an image leads to a noise cone in the 3D space. This effect and its correction is analyzed in Fig. 14. For this experiment, we calibrate a scene with a model house. Then we pick out two points of the model, put a Gaussian noise on their corresponding image points and estimate their pose separately. The two chosen points differ in their relative depth with respect to the image plane as can be seen in Fig. 14. Then we estimate the absolute image error. This means, the error measure is now connected to the observation of the pose in the image plane. The

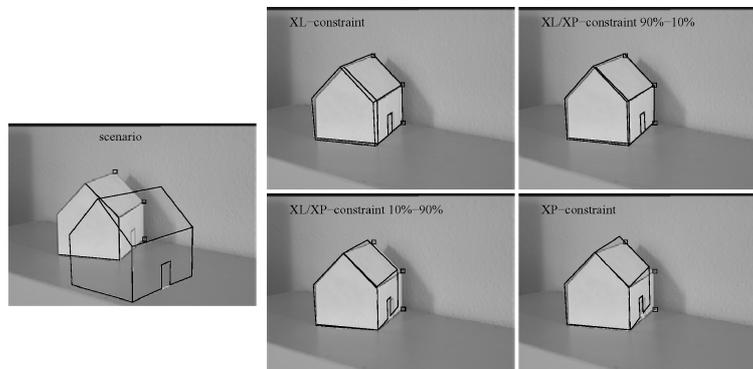


Figure 12. Different weights of constraints for pose estimation.

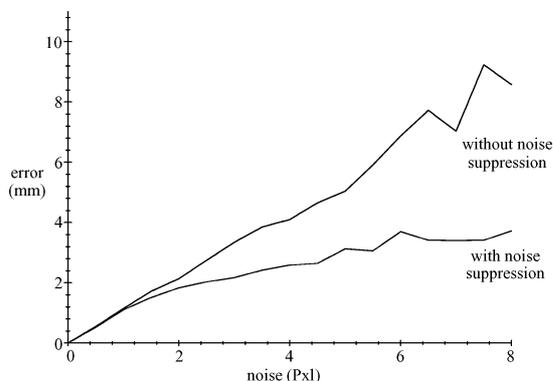


Figure 13. Comparison of the results with and without noise suppression.

graph in Fig. 14 shows the influence of the disturbed image points on the estimated pose and their effect on the image plane. The pixel noise of the image point is given on the  $x$ -axis and on the  $y$ -axis the absolute pixel error of the transformed projected object model compared with the ground truth is shown. It can be seen that, though the image points are disturbed in an equal manner, the result of the noisy *far* pixel is worse than the result of the noisy *near* pixel. This effect is often dis-

cussed as disadvantage of the 3D approach. But the possibility of noise adaptive use of the constraints is often neglected in this context. Since the constraint equations formalize the pose problem in an implicit manner, the constraints can be scaled with respect to their relative depth. This is shown in the third error curve of Fig. 14.

**4.2.3. Pose Estimation of Kinematic Chains.** In the next experiment (see e.g. Fig. 15), we use as more complex object model the RX-90 robot arm [40]. Figure 15 shows some examples of a sequence containing 42 images. In this image sequence the first joint is moving in 5 degrees steps from 0 to 25 degrees. Then the second joint is moving in 5 degrees steps from 0 to 60 degrees. This is also shown in Fig. 16. We estimate the pose of the robot and the angles of the kinematic chain via tracked points markers. Figure 16 shows the joint angles estimated and overlaid with the ground truth.<sup>2</sup> Small deviations can be recognized. Dependent on the position of the camera with respect to the object model and the location of the joints, the estimated angles differ around 0.5 to 3 degrees to the ground truth. In simulation environments (and ideal situations) we could prove that (for not degenerate cases) the parameters

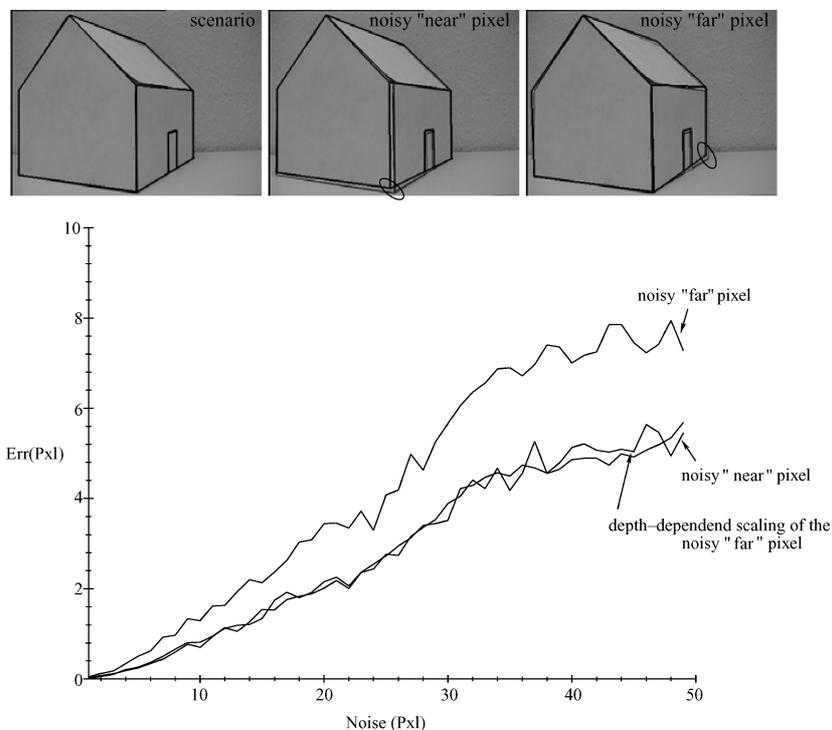


Figure 14. Depth dependence of the constraints.

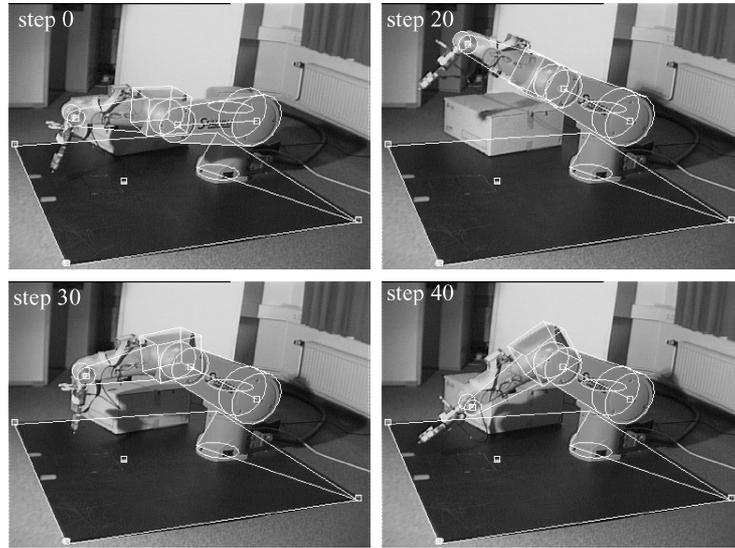


Figure 15. Images of a tracked robot arm taken from a sequence with 40 images.

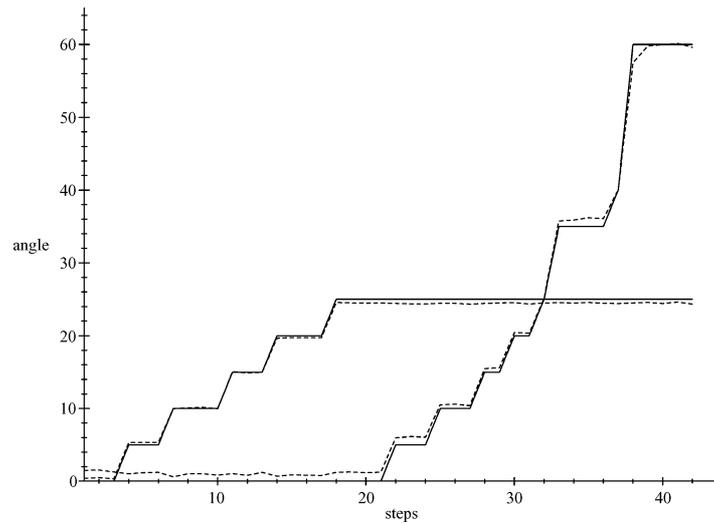


Figure 16. Joint angles estimated and overlaid with the ground truth. The solid lines show the ground truth and the dashed lines show the estimated values.

during the iterations converge against the ground truth. The errors we gain in these experiments are dependent on the calibration quality, the lens distortions and the accuracy of the color marker detection.

Figure 17 shows an other image sequence. There we visualize the stability of our algorithm in the context of moved color markers which corresponds to impossible kinematics of the robot. Two things can be seen. First, we model the geometry of the robot within our constraints and the model will not be distorted. Instead, the algorithm leads to a spatially best fit of the

model to the extracted image data. Second, we have no hierarchical approach for pose estimation of piecewise rigid objects as in [13, 43] but a pose estimation based on the model of a kinematic chain. Hierarchical pose estimation means that the pose problem is separated in subproblems which are solved sequentially. So first the whole pose (the base transformation) is estimated and then each joint angle separately. To ensure that the model is not distorted after the calculations the estimated values have to be constrained to the model in a second processing step. There are two main

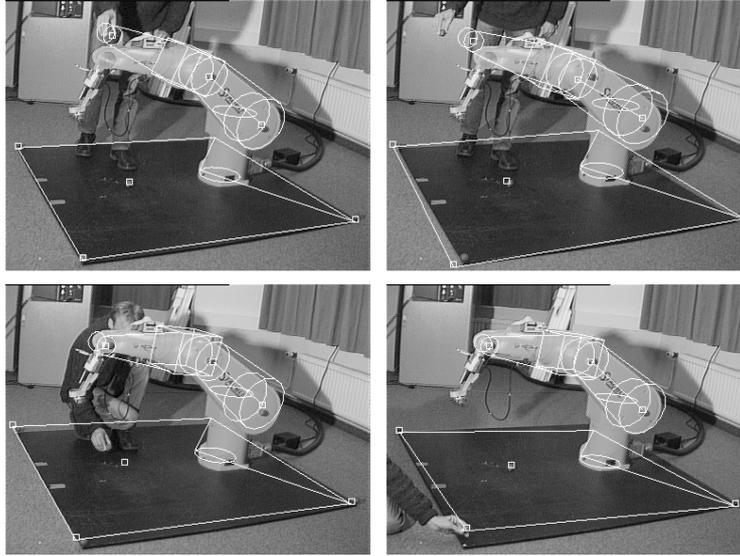


Figure 17. Stability example for disturbed color markers and visualization of the geometry of the robot which is modeled within the constraints.

arguments why we do not recommend this method: Firstly, the geometry of the whole object is not modeled within the constraint equations. That necessitates the second processing step to ensure no distorted model. This second processing step can be avoided by modeling a kinematic chain within the constraint equations, as is done in this work or by [4]. Second, each point of a kinematic chain contributes with two linear independent equations. Also the higher order points of a kinematic chain influence the result of the whole pose. This is strongly wanted in this context because only then all possible geometric information is used simultaneously and not neglected due to redundancy of the algorithm.

Figure 18 shows three example images taken from a project dealing with a *visual remote control* for the RX-90 robot. This means, a person with color markers attached to the person is tracked and the angles of the human arm movements are estimated and trans-

lated to the robot kinematics. So the robot *follows* the human movements and therefore the human is able to control the robot by its own movements. Color markers on the finger tips indicate the opening and closing of the gripper. This module is used for grasping and manipulation tasks as shown in Fig. 18. Since also a client-server package is implemented, image processing and tracking of the person is independent from the location of the robot. This enables a robot controlling though the person is not in the same room in which the robot is located. During the experiments, we let the image stream run via the university campus, which is 3 km away from the robot lab. The control system is shown in Fig. 19.

#### 4.2.4. Simultaneous Pose Estimation with Different Kinds of Correspondences.

This section concerns the use of more extended object concepts for pose estimation. In Fig. 20 pose estimation results of an object



Figure 18. Example images for visual remote controlling of the robot.

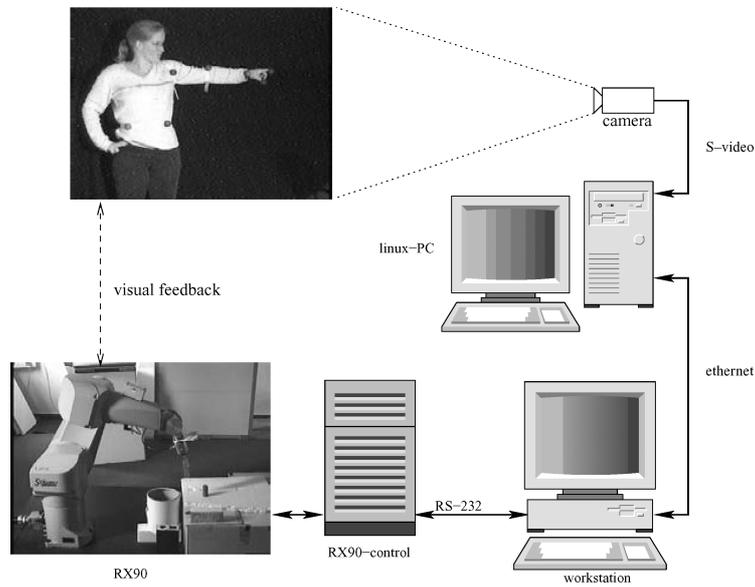


Figure 19. The system design for controlling the robot.

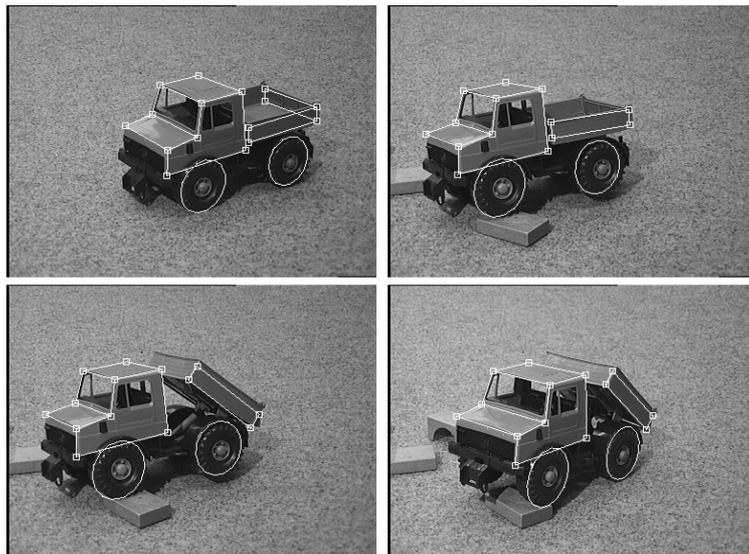


Figure 20. Pose estimation of an object, consisting of 3D points, lines, circles and kinematic chain segments.

containing points, lines, kinematic chains and circles are presented.

In the last experiment, we use a model which contains additionally a sphere, a prismatic and a revolute joint. The model is depicted in Fig. 21. Figure 22 shows some pose estimation results of the object model. Though we measured the size of the model by hand, the pose is accurate and also the joint parameters are good approximated. All information is arranged in

one linear system of equations, which leads to simultaneous solving of the pose parameters by using all different features.

### 5. Summary and Discussion

In this work several important topics for computer vision and robotics are discussed. First of all, we use the

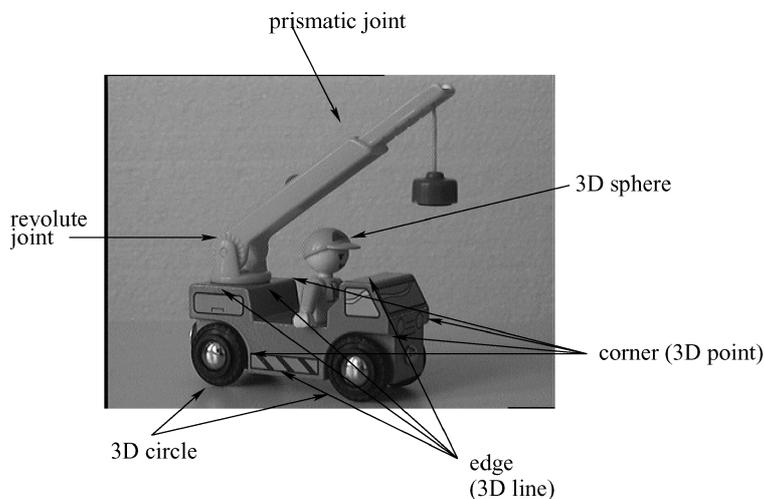


Figure 21. Object model, consisting of different entities.

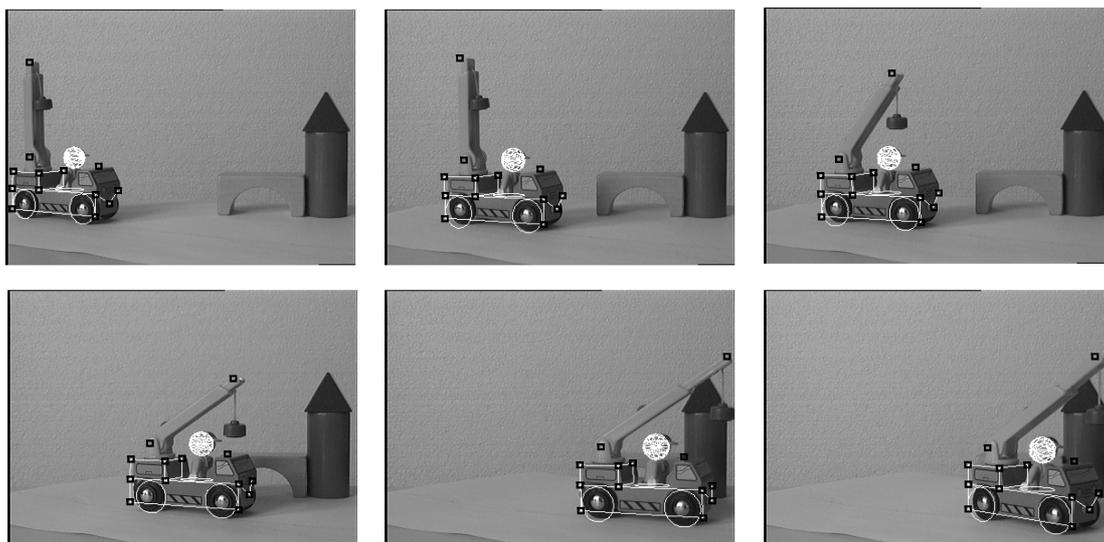


Figure 22. Pose estimation by using all types of model features.

foundations of Part I to deal with the pose estimation problem. The framework of conformal geometric algebra enables us to handle Euclidean, projective and conformal geometry by using suitable sub-algebras. We present an extended framework for pose estimation of object models, which consist of different types of entities, including points, lines, planes, circles, spheres and kinematic chains. We present efficient approaches to solve the pose estimation problem numerically by using all information simultaneously. Our experiments with monocular pose estimation of kinematic chains show that this is a first step to advantageously cope with

robot vision problems [27] in an advanced algebraic way. The algorithm for pose estimation of kinematic chains extends existing approaches, e.g. [4], since we do not build constraints in the image plane, but constraints in the 3D space. Furthermore, we are able to use a full perspective camera model in this context and not a scaled orthographic one. The equations of different constraints are put into one system of equations for estimating the parameters of the rigid body motion. Since each point of a kinematic chain contributes with two linearly independent equations, also the higher order points of kinematic chains influence

the result of the whole pose. This is also an extension to classical hierarchical approaches as presented in [13, 43].

Instead of using invariances as an explicit formulation of ideal geometry, we are using implicit formulations of geometry as geometric constraints. Since in our constraints spatial distance measures have to be minimized, we can quite easily deal with noisy image features, inexact calibrated cameras and noisy object model features. Because the optimization is performed with respect to the spatial distance in Euclidean space, the task of pose estimation is more simple in comparison to the minimization of distances on the manifold of rigid body motions as performed in [5, 41].

In particular the noise adaptive use of the constraints in this context is very interesting with respect to the design of behavior based [37] or learning robot systems. Several articles concerning the fusion of noisy data, e.g. [14], can be compared with our approach in that respect. But we are also interested to apply different kinds of entities with different reliabilities of measurements so that a system is able to adapt and to pick up the needed information by itself. Only few works exist so far which deal with this important topic for stable running systems, e.g. [17].

We implemented the sources in C++ [26] and are able to estimate the motion (and kinematic chain) parameters in real-time with 20 frames per second on a SUN Ultra 10 and gain 100 fps on a Linux 2GHz machine.

Our very recent work concerns extensions of this feature based pose approach and is presented in [33, 34]. There we use virtually coupled twists to yield a special family of curves (so-called 3D *cycloidal curves*) and extend this approach to general free-form contours.

### Acknowledgments

We would like to thank Oliver Granert, Daniel Grest, Jochen Koberstein, Norbert Krüger, Christian Perwass and Oliver Schmitz for fruitful discussions and hints for performing this work.

This work has been supported by DFG Graduiertenkolleg No. 357 and by EC Grant IST-2001-3422 (VISATEC).

### Notes

1. In the Eqs. (2.5), (2.8) and (4.3) the product sign  $X$  means simply juxtaposition (geometric product of factors on different lines).
2. Since the positioning accuracy of the robot arm is very good, we use the positioning values of the robot arm as ground truth.

### References

1. E. Bayro-Corrochano and D. Kähler, "Kinematics of robot manipulators in the motor algebra," pp. 473–490, 2001.
2. J.R. Beveridge, "Local search algorithms for geometric object recognition: Optimal correspondence and pose," Technical Report CS 93–5, University of Massachusetts, 1993.
3. W. Blaschke, *Kinematik und Quaternionen, Mathematische Monographien 4*, Deutscher Verlag der Wissenschaften, 1960.
4. C. Bregler and J. Malik, "Tracking people with twists and exponential maps," in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, Santa Barbara, California, 1998, pp. 8–15.
5. A. Chiuso and G. Picci, "Visual tracking of points as estimation on the unit sphere," in *The Confluence of Vision and Control*, Springer-Verlag, 1998, pp. 90–105.
6. K. Daniilidis, "Hand-eye calibration using dual quaternions," *Int. Journ. Robotics Res.*, Vol. 18, pp. 286–298, 1999.
7. J. Denavit and R.S. Hartenberg, "A kinematic notation for lower-pair mechanisms based on matrices," *ASME Journal of Applied Mechanics*, Vol. 22, pp. 215–221, 1955.
8. L. Dorst, "Honing geometric algebra for its use in the computer sciences," pp. 127–152, 2001 in [38].
9. O. Faugeras, "Stratification of three-dimensional vision: Projective, affine and metric representations," *Journal of Optical Society of America*, Vol. 12, No. 3, pp. 465–484, 1995.
10. M. Felsberg and G. Sommer, "The multidimensional isotropic generalization of quadrature filters in geometric algebra," in *2nd International Workshop on Algebraic Frames for the Perception-Action Cycle*, Springer-Verlag, 2000, LNCS 1888, pp. 175–185.
11. J. Gallier, *Geometric Methods and Applications for Computer Science and Engineering*, Springer-Verlag: New York, 2001.
12. W.E.L. Grimson, *Object Recognition by Computer*, The MIT Press, Cambridge, MA, 1990.
13. A. Hauck, S. Lanser, and C. Zierl, "Hierarchical recognition of articulated objects from single perspective views," *IEEE: Computer Vision and Pattern Recognition*, Puerto Rico, pp. 870–876, 1997.
14. Y. Hel-Or and M. Werman, "Pose estimation by fusing noisy data of different dimensions," *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, Vol. 17, No. 2, pp. 195–201, 1995.
15. D. Hestenes, H. Li, and A. Rockwood, "New algebraic tools for classical geometry," pp. 3–23, 2001 in [38].
16. D. Hestenes and R. Ziegler "Projective geometry with Clifford algebra," *Acta Applicandae Mathematicae*, Vol. 23, pp. 25–63, 1991.
17. J.R. Holt and A.N. Netravali, "Uniqueness of solutions to structure and motion from combinations of point and line correspondences," *Journal of Visual Communication and Image Representation*, Vol. 7, No. 2, pp. 126–136, 1996.
18. H.H. Homer, "Pose determination from line-to-plane correspondences: Existence condition and closed-form solutions," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 13, No. 6, pp. 530–541, 1991.
19. R. Horaud, T.Q. Phong, and P.D. Tao, "Object pose from 2-d to 3-d point and line correspondences," *International Journal of Computer Vision*, Vol. 15, pp. 225–243, 1995.

20. H. Li, D. Hestenes, and A. Rockwood, "Generalized homogeneous coordinates for computational geometry," pp. 27–52, 2001 in [38].
21. A. Lorusso, D.W. Eggert, and R.B. Fisher, "A comparison of four algorithms for estimating 3D rigid transformations," *Machine Vision and Applications*, Vol. 9, Nos. 5/6, pp. 272–290, 1997.
22. D.G. Lowe, "Solving for the parameters of object models from image descriptions," in *Proc. ARPA Image Understanding Workshop*, 1980, pp. 121–127.
23. D.G. Lowe, "Three-dimensional object recognition from single two-dimensional images," *Artificial Intelligence*, Vol. 31, No. 3, pp. 355–395, 1987.
24. R.M. Murray, Z. Li, and S.S. Sastry, *A Mathematical Introduction to Robotic Manipulation*, CRC Press, 1994.
25. T. Needham, *Visual Complex Analysis*, Oxford University Press, 1997.
26. PACLib (Homepage of the Kiel Perception-Action-Components Library). <http://www.ks.informatik.uni-kiel.de/~paclib/>
27. J. Pauli, "Development of camera-equipped robot systems," Technical Report 9904, Christian-Albrechts-Universität zu Kiel, Institut für Informatik und Praktische Mathematik, 2000.
28. C. Perwass and J. Lasenby, "A novel axiomatic derivation of geometric algebra," Technical Report CUED/F—INFENG/TR.347, Cambridge University Engineering Department, 1999.
29. W.H. Press, B.P. Flannery, S.A. Teukolsky, and W.T. Vetterling, *Numerical Recipes*, C. Cambridge University Press, 1993.
30. B. Rosenhahn, O. Granert, and G. Sommer, "Monocular pose estimation of kinematic chains," in *Applied Geometric Algebras for Computer Science and Engineering*, L. Dorst, C. Doran, and J. Lasenby (Eds.), Birkhäuser Verlag, 2001, pp. 373–383.
31. B. Rosenhahn, N. Krüger, T. Rabsch, and G. Sommer, "Tracking with a novel pose estimation algorithm," in *International Workshop on Robot Vision*, R. Klette, S. Peleg, and G. Sommer (Eds.), Springer-Verlag Heidelberg, 2001, LNCS 1998, pp. 9–19.
32. B. Rosenhahn and J. Lasenby, "Constraint equations for 2D-3D pose estimation in conformal geometric algebra," Technical Report CUED/F—INFENG/TR.396, Cambridge University Engineering Department, 2000.
33. B. Rosenhahn, C. Perwass, and G. Sommer, "Pose estimation of 3D free-form contours," Technical Report 0207, Christian-Albrechts-Universität zu Kiel, Institut für Informatik und Praktische Mathematik, 2002.
34. B. Rosenhahn, C. Perwass, and G. Sommer, "Pose estimation of 3D free-form contours in conformal geometry," in *Proceedings of Image and Vision Computing (IVCNZ)* D. Kenwright (Ed.), New Zealand, 2002, pp. 29–34.
35. B. Rosenhahn and G. Sommer, "Pose estimation in conformal geometric algebra. Part I: The stratification of mathematical spaces," *Journal of Mathematical Imaging and Vision*, Vol. 22, No. 1 pp. 27–48, 2005.
36. F. Shevlin, "Analysis of orientation problems using Plücker lines," *International Conference on Pattern Recognition, Brisbane*, Vol. 1, pp. 685–689, 1998.
37. G. Sommer, "Algebraic aspects of designing behavior based systems," in *Algebraic Frames for the Perception-Action Cycle, AFPAC'97*, G. Sommer and J.J. Koenderink (Eds.), Springer-Verlag, Heidelberg, 1997, LNCS 1315, pp. 1–28.
38. G. Sommer (Ed.), *Geometric Computing with Clifford Algebras*, Springer-Verlag, Heidelberg, 2001.
39. G. Sommer, B. Rosenhahn, and Y. Zhang, "Pose estimation using geometric constraints," in *Multi-Image Search and Analysis*, R. Klette, Th. Huang, and G. Gimmel'farb (Eds.), Springer-Verlag, Heidelberg, 2001, LNCS 2032, pp. 153–170.
40. Stäubli Roboter RX90-CS7 Instruction Manual D280.190.12.C, November, 1992.
41. A. Ude, "Filtering in a unit quaternion space for model-based object tracking," *Robotics and Autonomous Systems*, Vol. 28, Nos. 2/3, pp. 163–172, 1999.
42. M.W. Walker and L. Shao, "Estimating 3-d location parameters using dual number quaternions," *CVGIP: Image Understanding*, Vol. 54, No. 3, pp. 358–367, 1991.
43. S. Weik and C.-E. Liedtke, "Hierarchical 3D pose estimation for articulated human body models," in *International Workshop on Robot Vision*, R. Klette, S. Peleg, and G. Sommer (Eds.), Springer-Verlag Heidelberg, 2001, LNCS 1998, pp. 27–34.



**Bodo Rosenhahn** gained his diploma degree in Computer Science in 1999. Since then he has been pursuing his Ph.D. at the Cognitive Systems Group, Institute of Computer Science, Christian-Albrechts University Kiel, Germany. He is working on geometric applications of Clifford algebras in computer vision.



**Prof. Dr. Gerald Sommer** received a diploma degree in physics from the Friedrich-Schiller-Universität Jena, Germany, in 1969, a Ph.D. degree in physics from the same university in 1975, and a habilitation degree in engineering from the Technical University Ilmenau, Germany, in 1988. Since 1993 he is leading the research group Cognitive Systems at the Christian-Albrechts-Universität Kiel, Germany. Currently he is also the scientific coordinator of the VISATEC project.