Robotica (2007) volume 25, pp. 43–61. © 2006 Cambridge University Press
doi:10.1017/S0263574706002980 Printed in the United Kingdom

Differential and inverse kinematics of robot devices using conformal geometric algebra

Eduardo Bayro-Corrochano* and Julio Zamora-Esquivel

Electrical Engineering and Computer Science Department, GEOVIS Laborator, Centro de Investigación y de Estudios Avanzados, Guadalajara, Jalisco 44550, Mexico
E-mail: jzamora@gdl.cinvestav.mx

(Received in Final Form: June 12, 2006, First published online: August 29, 2006)

SUMMARY
In this paper, the authors use the conformal geometric algebra in robotics. This paper computes the inverse kinematics of a robot arm and the differential kinematics of a pan–tilt unit using a language of spheres showing how we can simplify the complexity of the computations.

This work introduces a new geometric Jacobian in terms of bivectors, which is by far more effective in its representation as the standard Jacobian because its derivation is done in terms of the projections of the involved points onto the line axes. Furthermore, unlike the standard formulation, our Jacobian can be used for any kind of robot joints.

In this framework, we deal with various tasks of three-dimensional (3D) object manipulation, which is assisted by stereo-vision. All these computations are carried out using real images captured by a robot binocular head, and the manipulation is done by a five degree of freedom (DOF) robot arm mounted on a mobile robot. In addition to this, we show a very interesting application of the geometric Jacobian for differential control of the binocular head. We strongly believe that the framework of conformal geometric algebra can generally be of great advantage for visually guided robotics.

KEYWORDS: Computer vision; Clifford (geometric) algebra; Projective and affine geometry; Spheres projective geometry; Incidence algebra; 3D rigid motion; Directed distance; Inverse kinematics; Differential geometry; Robot 3D object manipulation; Stereo systems; Smooth control of binocular heads; Visually guided robotics.

1. Introduction

In the literature after the sixties, we find a variety of mathematical systems used for solving problems in general robotics, which we will review briefly. Denavit and Hartenberg[10] introduced the widely used kinematic notation for lower pair mechanisms based on matrix algebra, Walker[24] used the epsilon algebra for the treatment of the manipulator kinematics, Gu and Luh[12] utilized dual-matrices for computing the Jacobians useful for kinematics and robot dynamics, and Pennock and Yang[20] derived closed-form solutions for the inverse kinematics problem for various types of robot manipulators employing dual-matrices. Similarly, McCarthy[19] used the dual form of the Jacobian for the analysis of multilinks. Funda and Paul[11] gave a detailed computational analysis of the use of screw transformations in robotics. They explained that since the dual quaternion can represent the rotation and translation transformations simultaneously, it is more effective than the unit quaternion formalism for dealing with the kinematics of robot chains. Many practitioners use a quaternion for 3D rigid transformations for representing the 3D rotation and simply a 3D translation vector. It can easily be shown that this kind of representation is a nonlinear function that leads to nonlinear algorithms for a sequential estimation of rotation and translation. Kim and Kumar[17] computed a closed-form solution of the inverse kinematics of a six DOF robot manipulator in terms of line transformations using dual quaternions. Aspragathos and Dimitros[2] confirmed once again that the use of dual quaternion and Lie algebra in robotics were overseen so far, and that their use helps to reduce the number of representation parameters.

In the field of computer vision, although in a different context as robotics, we can find similar representation formalisms in various types of applications like motion estimation, pose and 3D structure recognition, tracking and visual servoying. In most of the methods, the rotation and translation transformations were represented separately using either matrices or quaternions (see the survey of Sabata and Aggarwal[21]). The disadvantage of separately representing these components is that for solving the problems, nonlinear methods are often required. In the case of the so-called hand–eye calibration problem, for the computation of the rotation axis and angle, several authors considered[22,23] the use of quaternions[9] and a canonical matrix representation.[18] Chen,[8] using the matrix screw theory, found as key invariant of the screw between the two 3D axes that the rotation angle and the translation along the screw axis remained constant. For solving the hand–eye calibration in a linear manner, Bayro-Corrochano *et al.*[4] used a Clifford algebra of lines called the motor algebra. In other applications, the authors successfully applied dual quaternions; e.g., Walker *et al.*[24] for estimating the 3D location, and twists and exponential maps like Bregler and Malik[7] for tracking the kinematic chains of moving objects or persons. For solving the hand–eye problem for visual line tracking, Andreff[1] used a matrix approach for a sort of algebra of screws.

* Corresponding author. edb@gdl.cinvestav.mx

We can see in all these mathematical approaches in the field of robotics, computer vision, and visual-guided robotics that basically the authors have taken into account two key aspects: the obvious use of dual numbers and the representation of the screw transformations in terms of matrices or dual quaternions. In this regard, in ref. 5 we were concerned with the extension of the representation capabilities of the dual numbers, particularly using the motor algebra beside the point and line representation we are able to model the motion of planes. This widened up the possibilities, for example, by the modeling of the motion of the basic geometric objects referred to frames attached to the robot manipulator, which according to circumstances greatly simplify the complexity of the problem preserving the underlying geometry. In this past paper after giving the model of prismatic and revolute transformations of a robot manipulator using points, lines, and planes, we solved the direct and inverse kinematics of robot manipulators. Using the motion of points, lines, and planes in terms of motors, we present constraints for a simple grasping task. This paper clearly shows the advantages of the use of representations in motor algebra for solving problems related to robot manipulators.

However, we are still interested in offering to the community, a much more general mathematical framework than the motor algebra. That is why we later introduced[6] the conformal geometric algebra as a flexible language for treating a variety of problems like scene understanding, robot navigation, visual servoying, haptics, and robot object manipulation. Unlike the standard projective geometry used routinely in computer vision and visual servoying, in conformal geometric algebra, by assuming that the camera is calibrated, we can deal simultaneously with incidence algebra operations (meet and join) and conformal transformations represented effectively using spinors. In this regard, this framework appears promising for dealing with kinematics, dynamics, and projective geometry problems without the need to abandon the mathematical system (unlike the current approaches). We will show that this mathematical framework keeps our intuition and insight of the geometry of the problem at hand, making the development of the algorithms easy in a truly geometrical sense. It also helps us to considerably reduce the computational burden of the problems.

In this paper, we compute the inverse kinematics of a robot arm and a robot pan–tilt unit using a language of spheres, showing how the complexity of the computations can be simplified. This work is an extension of our previous approach using the motor algebra for computing the inverse kinematics in terms of points, lines, and planes,[5] and other works using points and lines[17] or dual orthogonal matrices.[19] One important contribution of this work is the introduction of a new geometric Jacobian in terms of bivectors, which is by far more effective in its representation as the standard Jacobian due to its easy derivation that is done in terms of the projections of the involved points onto the line axes of the robot joints. Unlike the standard formulation, our Jacobian is more general, because it can be used for any kind of robot joints. In this framework, we also deal with the 3D object manipulation, which is assisted by stereo-vision. We consider the following tasks: touching a point, following the interaction of two planes, following a spherical path, and grasping an object. All these computations are carried out using real images captured by a robot binocular head, and the manipulation is done by a five DOF robot arm mounted on a mobile robot.

The organization of the paper is as follows: Section 2 presents a brief introduction about conformal geometric algebra. Section 3 describes the n-dimensional affine plane. Section 4 explains the transformations in conformal geometric algebra required for this paper. Section 5 presents the differential kinematics and the derivation of geometric Jacobian. Section 6 shows the computation of the inverse kinematics of robot devices using conformal geometry. Section 7 explains the visual Jacobian useful for closing the loop between the visual and the mechanical worlds. Section 8 shows the geometric techniques for the following up of geometric visual primitives, and Section 9 computes the differential kinematic control of a pan–tilt unit. Section 10 includes the geometric strategies for object manipulation. Section 11 provides conclusions.

In this paper, vectors in a 3D Euclidean space have been represented in bold and lower case. Vectors of geometric algebra have been written using lower-case letters (slant and bold except for basis multivectors), and slant and upper case have been used to denote multivectors in general.

2. Geometric Algebra

In general, a geometric algebra G_n is an n-dimensional vector space V^n over the reals. We also denote with $G_{p,q,r}$ a geometric algebra over $V^{p,q,r}$, where p, q, r denote the signature p, q, r of the algebra. If $p \neq 0$ and $q = r = 0$, the metric is Euclidean G_n; if just $r = 0$, the metric is pseudoeuclidean $G_{p,q}$; if none of them are zero, the metric is degenerate. See refs. 6 and 15 for a more detailed introduction to the conformal geometric algebra.

We have used the letter e to denote the vector basis e_i. In a geometric algebra $G_{p,q,r}$, the geometric product of two basis vectors is defined as

$$e_i e_j = \begin{cases} 1 & \text{for } i = j \in 1, \ldots, \mathtt{p} \\ -1 & \text{for } i = j \in \mathtt{p}+1, \ldots, \mathtt{p}+\mathtt{q} \\ 0 & \text{for } i = j \in \mathtt{p}+\mathtt{q}+1, \ldots, \mathtt{p}+\mathtt{q}+\mathtt{r} \\ e_i \wedge e_j & \text{for } i \neq j. \end{cases} \tag{1}$$

2.1. Conformal geometric algebra

The geometric algebra of 3D Euclidean space $\mathcal{G}_{3,0,0}$ has a point basis, and the motor algebra $\mathcal{G}_{3,0,1}$ has a line basis. In the latter geometric algebra, the lines expressed in terms of Plücker coordinates can be used to represent points and planes as well.[4] The reader can find a comparison of representations of points, lines, and planes using $\mathcal{G}_{3,0,0}$ and $\mathcal{G}_{3,0,1}$ in ref. 4.

Interesting enough in the case of the conformal geometric algebra, we find that the unit element is the sphere that allows us to represent the other geometric primitives in its terms. To see how this is possible, we begin giving an introduction in conformal geometric algebra following the same formulation presented in ref. 15, and we show how the Euclidean vector space $\mathbb{R}^n$ is represented in $\mathbb{R}^{n+1,1}$. This space has an

orthonormal vector basis given by $\{e_1, \ldots, e_n, e_+, e_-\}$ with the properties

$$e_i^2 = 1, \quad i = 1, \ldots, n \tag{2}$$

$$e_\pm^2 = \pm 1 \tag{3}$$

$$e_i \cdot e_+ = e_i \cdot e_- = e_+ \cdot e_- = 0, \quad i = 1, \ldots, n. \tag{4}$$

Note that this basis is not written in bold.

A *null basis* $\{e_0, e_\infty\}$ can be introduced by

$$e_0 = \frac{e_- - e_+}{2} \tag{5}$$

$$e_\infty = e_- + e_+ \tag{6}$$

with the properties

$$e_0^2 = e_\infty^2 = 0, \quad e_\infty \cdot e_0 = -1. \tag{7}$$

A unit pseudoscalar $E \in \mathbb{R}^{1,1}$ that represents the Minkowski plane is defined by

$$\boldsymbol{E} = e_\infty \wedge e_0 = e_+ \wedge e_- = e_+ e_-. \tag{8}$$

Euclidean points $\mathbf{x}_e \in \mathbb{R}^n$ can be represented in $\mathbb{R}^{n+1,1}$ in a general way as

$$\begin{aligned} \boldsymbol{x}_c &= (\boldsymbol{x}_c \wedge \boldsymbol{E})\boldsymbol{E} + (\boldsymbol{x}_c \cdot \boldsymbol{E})\boldsymbol{E} = \mathbf{x}_e + e_0 + \frac{1}{2}(k_1 + k_2)e_\infty \\ &= \mathbf{x}_e + \frac{1}{2}\mathbf{x}_e^2 e_\infty + e_0. \end{aligned} \tag{9}$$

We can gain further insight into the geometrical meaning of the null vectors by analyzing Eq. (9). For instance by setting $\mathbf{x}_e = 0$, we find that e_0 represents the origin of $\mathbb{R}^n$ (hence the name). Similarly, dividing this equation by $\boldsymbol{x}_c \cdot e_0 = -\frac{1}{2}\mathbf{x}_e^2$ gives

$$\begin{aligned} \frac{\boldsymbol{x}_c}{\boldsymbol{x}_c \cdot e_0} &= -\frac{2}{\mathbf{x}_e^2}\left(\mathbf{x}_e + \frac{1}{2}\mathbf{x}_e^2 e_\infty + e_0\right) \\ &= -\frac{2\mathbf{x}_e^2}{\mathbf{x}_e^2}\left(\frac{1}{\mathbf{x}_e} + \frac{1}{2}e_\infty + \frac{e_0}{\mathbf{x}_e^2}\right) \\ &= -2\left(\frac{1}{\mathbf{x}_e} + \frac{1}{2}e_\infty + \frac{e_0}{\mathbf{x}_e^2}\right) \underset{\mathbf{x}_e \to \infty}{\longrightarrow} e_\infty. \end{aligned} \tag{10}$$

Thus, we conclude that e_∞ represents the point at infinity.

2.2. Spheres and planes

The equation of a sphere of radius ρ centered at point $\mathbf{p}_e \in \mathbb{R}^n$ can be written as

$$(\mathbf{x}_e - \mathbf{p}_e)^2 = \rho^2. \tag{11}$$

Since $\boldsymbol{x}_c \cdot \boldsymbol{y}_c = -\frac{1}{2}(\mathbf{x}_e - \mathbf{y}_e)^2$, we can rewrite the formula given above in terms of homogeneous coordinates as

$$\boldsymbol{x}_c \cdot \boldsymbol{p}_c = -\frac{1}{2}\rho^2. \tag{12}$$

Since $\boldsymbol{x}_c \cdot e_\infty = -1$, we can factor the expression given above to

$$\boldsymbol{x}_c \cdot \left(\boldsymbol{p}_c - \frac{1}{2}\rho^2 e_\infty\right) = 0 \tag{13}$$

which finally yields the simplified equation for the sphere as

$$\boldsymbol{x}_c \cdot \boldsymbol{s} = 0 \tag{14}$$

where

$$\boldsymbol{s} = \boldsymbol{p}_c - \frac{1}{2}\rho^2 e_\infty = \mathbf{p}_e + e_0 + \frac{\mathbf{p}_e^2 - \rho^2}{2}e_\infty \tag{15}$$

is the equation of the sphere (note from this equation that a point is just a sphere with zero radius). The vector s has the properties

$$\boldsymbol{s}^2 = \rho^2 > 0 \tag{16}$$

$$e_\infty \cdot \boldsymbol{s} = -1. \tag{17}$$

From these properties, we conclude that the sphere s is a point lying on the hyperplane, but *outside* the null cone. In particular, all points on the hyperplane outside the horosphere determine spheres with positive radius, points lying on the horosphere define spheres of zero radius (i.e., points), and points lying inside the horosphere have imaginary radius. Finally, note that spheres of the same radius form a surface that is parallel to the horosphere.

Alternatively, spheres can be dualized and represented as $(n+1)$-vectors $\boldsymbol{s}^* = \boldsymbol{s}I^{-1}$. Since

$$\widetilde{I} = (-1)^{\frac{1}{2}(n+2)(n+1)} I = -I^{-1} \tag{18}$$

we can express the constraints of Eqs. (16) and (17) as

$$\begin{aligned} \boldsymbol{s}^2 &= -\widetilde{\boldsymbol{s}^*}\boldsymbol{s}^* = \rho^2 \\ e_\infty \cdot \boldsymbol{s} &= e_\infty \cdot (\boldsymbol{s}^* I) = (e_\infty \wedge \boldsymbol{s}^*)I = -1. \end{aligned} \tag{19}$$

The equation for the sphere now becomes

$$\boldsymbol{x}_c \wedge \boldsymbol{s}^* = 0. \tag{20}$$

The advantage of the dual form is that the sphere can be directly computed from four points (in 3D) as

$$\boldsymbol{s}^* = \boldsymbol{x}_{c_1} \wedge \boldsymbol{x}_{c_2} \wedge \boldsymbol{x}_{c_3} \wedge \boldsymbol{x}_{c_4}. \tag{21}$$

If we replace one of these points for the point at infinity, we get

$$\pi^* = \boldsymbol{x}_{c_1} \wedge \boldsymbol{x}_{c_2} \wedge \boldsymbol{x}_{c_3} \wedge e_\infty. \tag{22}$$

Developing the products, we get

$$\begin{aligned} \pi^* &= \boldsymbol{x}_{c_3} \wedge \boldsymbol{x}_{c_1} \wedge \boldsymbol{x}_{c_2} \wedge e_\infty \\ &= \mathbf{x}_{e_3} \wedge \mathbf{x}_{e_1} \wedge \mathbf{x}_{e_2} \wedge e_\infty + ((\mathbf{x}_{e_3} - \mathbf{x}_{e_1}) \wedge (\mathbf{x}_{e_2} - \mathbf{x}_{e_1}))\boldsymbol{E} \end{aligned} \tag{23}$$

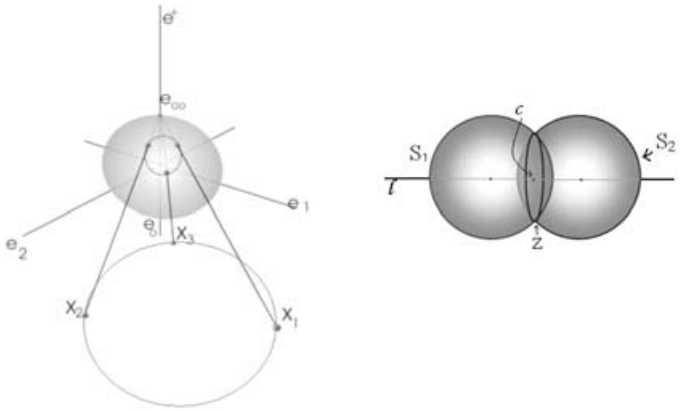

Fig. 1. (a) Circle computed using three points. Note the stereographic projection. (b) Circle computed using the meet of two spheres.

which is the equation of the plane passing through the points $\mathbf{x}_{e_1}$, $\mathbf{x}_{e_2}$, and $\mathbf{x}_{e_3}$. We can easily see that $\mathbf{x}_{e_1} \wedge \mathbf{x}_{e_2} \wedge \mathbf{x}_{e_3}$ is a scalar representing the volume of the parallelepiped with sides $\mathbf{x}_{e_1}$, $\mathbf{x}_{e_2}$, and $\mathbf{x}_{e_3}$. Also, since $(\mathbf{x}_{e_1} - \mathbf{x}_{e_2})$ and $(\mathbf{x}_{e_3} - \mathbf{x}_{e_2})$ are two vectors on the plane, the expression $((\mathbf{x}_{e_1} - \mathbf{x}_{e_2}) \wedge (\mathbf{x}_{e_3} - \mathbf{x}_{e_2}))$ is the normal to the plane. Therefore, planes are spheres passing through the point at infinity.

2.3. Geometric identities, duals, and incidence algebra operations

A circle $\boldsymbol{z}$ can be regarded as the intersection of two spheres $\boldsymbol{s}_1$ and $\boldsymbol{s}_2$. This means that for each point on the circle $\boldsymbol{x}_c \in \boldsymbol{z}$ they lie on both spheres as well as $\boldsymbol{x}_c \in \boldsymbol{s}_1$ and $\boldsymbol{x}_c \in \boldsymbol{s}_2$. Assuming that $\boldsymbol{s}_1$ and $\boldsymbol{s}_2$ are linearly independent, we can write for $\boldsymbol{x}_c \in \boldsymbol{z}$

$$(\boldsymbol{x}_c \cdot \boldsymbol{s}_1)\boldsymbol{s}_2 - (\boldsymbol{x}_c \cdot \boldsymbol{s}_2)\boldsymbol{s}_1 = x_c \cdot (\boldsymbol{s}_1 \wedge \boldsymbol{s}_2) = \boldsymbol{x}_c \cdot \boldsymbol{z} = 0. \quad (24)$$

The result tells us that since $\boldsymbol{x}_c$ lies on both the spheres, $\boldsymbol{z} = (\boldsymbol{s}_1 \wedge \boldsymbol{s}_1)$ should be the intersection of the spheres or a circle. It is easy to see that the intersection with a third sphere leads to a point pair. We have derived algebraically that the wedge of two linearly independent spheres yields to their intersecting circle (Fig. 1), this topological relation between the two spheres can also be conveniently described using the dual of the meet operation, namely

$$\boldsymbol{z} = (\boldsymbol{z}^*)^* = (\boldsymbol{s}_1^* \vee \boldsymbol{s}_2^*)^* = \boldsymbol{s}_1 \wedge \boldsymbol{s}_2. \quad (25)$$

This new equation shows that the dual of a circle can be computed via the meet of the two spheres in their dual form. This equation geometrically confirms our previous algebraic computation of Eq. (24).

The dual form of the circle (in 3D) can be expressed by three points lying on it [Fig. 1(a)] as

$$\boldsymbol{z}^* = \boldsymbol{x}_{c_1} \wedge \boldsymbol{x}_{c_2} \wedge \boldsymbol{x}_{c_3}. \quad (26)$$

Similar to the case of planes show in Eq. (22), lines can be defined by circles passing through the point at infinity as

$$\boldsymbol{l}^* = \boldsymbol{x}_{c_1} \wedge \boldsymbol{x}_{c_2} \wedge e_\infty. \quad (27)$$

This can be demonstrated by developing the wedge products, as in the case of the planes, to yield

$$\boldsymbol{x}_{c_1} \wedge \boldsymbol{x}_{c_2} \wedge e_\infty = \mathbf{x}_{e_1} \wedge \mathbf{x}_{e_2} \wedge e_\infty + (\mathbf{x}_{e_2} - \mathbf{x}_{e_1}) \wedge \boldsymbol{E}, \quad (28)$$

from where it is evident that the expression $\boldsymbol{x}_{e_1} \wedge \boldsymbol{x}_{e_2}$ is a bivector representing the plane where the line is contained, and $(\boldsymbol{x}_{e_2} - \boldsymbol{x}_{e_1})$ is the direction of the line.

The dual of a point $\boldsymbol{p}$ is a sphere $\boldsymbol{s}$. The intersection of four spheres yields a point [Fig. 2(b)]. The dual relationships between a point and its dual, the sphere, are

$$\boldsymbol{s}^* = \boldsymbol{p}_1 \wedge \boldsymbol{p}_2 \wedge \boldsymbol{p}_3 \wedge \boldsymbol{p}_4 \leftrightarrow \boldsymbol{p}^* = \boldsymbol{s}_1 \wedge \boldsymbol{s}_2 \wedge \boldsymbol{s}_3 \wedge \boldsymbol{s}_4 \quad (29)$$

where the points are denoted as $\boldsymbol{p}_i$ and the spheres $\boldsymbol{s}_i$ for $i = 1, 2, 3, 4$.

A summary of the basic geometric entities and their duals ispresented in Table I.

There is another very useful relationship between an $(r-2)$ -dimensional sphere $\boldsymbol{A}_r$ and the sphere $\boldsymbol{s}^*$ (computed as the dual of a point $\boldsymbol{s}$). If from the sphere $\boldsymbol{A}_r$ we can compute

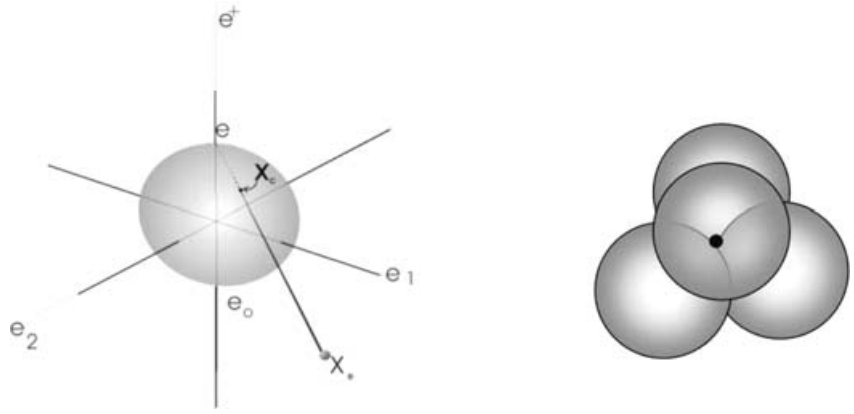

Fig. 2. (a) Conformal point generated by projecting a point of the affine plane to the unit sphere. (b) Point generated by the meet of four spheres.

Table I. Entities in conformal geometric algebra.

Entity	Representation	Grade	Dual representation	Grade
Sphere	$\boldsymbol{s} = \mathbf{p} + \frac{1}{2}(\mathbf{p}^2 - \rho^2)e_\infty + e_0$	1	$\boldsymbol{s}^* = \boldsymbol{a} \wedge \boldsymbol{b} \wedge \boldsymbol{c} \wedge \boldsymbol{d}$	4
Point	$\boldsymbol{x} = \mathbf{x} + \frac{1}{2}\mathbf{x}^2 e_\infty + e_0$	1	$\boldsymbol{x}^* = \boldsymbol{s}_1 \wedge \boldsymbol{s}_2 \wedge \boldsymbol{s}_3 \wedge \boldsymbol{s}_4$	4
Plane	$\pi = \boldsymbol{n} I_E - d e_\infty$ $\boldsymbol{n} = (\boldsymbol{a} - \boldsymbol{b}) \wedge (\boldsymbol{a} - \boldsymbol{c})$ $d = (\boldsymbol{a} \wedge \boldsymbol{b} \wedge \boldsymbol{c}) I_E$	1	$\pi^* = e_\infty \wedge \boldsymbol{a} \wedge \boldsymbol{b} \wedge \boldsymbol{c}$	4
Line	$\boldsymbol{L} = \pi_1 \wedge \pi_2$ $\boldsymbol{L} = \boldsymbol{n} I_E - e_\infty \boldsymbol{m} I_E$ $\boldsymbol{n} = (\boldsymbol{a} - \boldsymbol{b})$ $\boldsymbol{m} = (\boldsymbol{a} \wedge \boldsymbol{b})$	2	$\boldsymbol{L}^* = e_\infty \wedge \boldsymbol{a} \wedge \boldsymbol{b}$	3
Circle	$z = \boldsymbol{s}_1 \wedge \boldsymbol{s}_2$	2	$z^* = \boldsymbol{a} \wedge \boldsymbol{b} \wedge \boldsymbol{c}$	3
Point pair	$\boldsymbol{PP} = \boldsymbol{s}_1 \wedge \boldsymbol{s}_2 \wedge \boldsymbol{s}_3$ $\boldsymbol{PP} = \boldsymbol{s} \wedge \boldsymbol{L}$	3 2	$\boldsymbol{PP}^* = \boldsymbol{a} \wedge \boldsymbol{b}$	2

the hyperplane $\boldsymbol{A}_{r+1} \equiv \boldsymbol{e}_\infty \wedge \boldsymbol{A}_r \neq 0$, we can express the meet between the dual of the point $\boldsymbol{s}$ (a sphere) and the hyperplane $\boldsymbol{A}_{r+1}$ getting the sphere $\boldsymbol{A}_r$ of one dimension lower

$$(-1)^\epsilon \boldsymbol{s}^* \cap \boldsymbol{A}_{r+1} = (\boldsymbol{s}^* \boldsymbol{I}) \cdot \boldsymbol{A}_{r+1} = \boldsymbol{s} \boldsymbol{A}_{r+1} = \boldsymbol{A}_r. \quad (30)$$

This result tells us an interesting relationship: the sphere $\boldsymbol{A}_r$ and the hyperplane $\boldsymbol{A}_{r+1}$ are related via the point $\boldsymbol{s}$ (dual of the sphere $\boldsymbol{s}^*$); thus, we then rewrite Eq. (30) as follows:

$$\boldsymbol{s} = \boldsymbol{A}_r \boldsymbol{A}_{r+1}^{-1}. \quad (31)$$

Using Eq. (31) and given the plane π (A_{r+1}) and the circle z ($\boldsymbol{A}_r$), we can compute the sphere as [Fig. 3(a)]

$$\boldsymbol{s} = z \pi^{-1}. \quad (32)$$

Similarly, we can compute another important geometric relationship called the *pair of points* using Eq. (31) directly as

$$\boldsymbol{s} = \boldsymbol{P} \boldsymbol{P} \boldsymbol{L}^{-1}. \quad (33)$$

Using this result, given the line $\boldsymbol{L}$ and the sphere $\boldsymbol{s}$, we can compute the pair of points $\boldsymbol{PP}$ [Fig. 3(b)] as

$$\boldsymbol{PP} = \boldsymbol{sL} = \boldsymbol{s} \wedge \boldsymbol{L}. \quad (34)$$

3. The 3D Affine Plane

We have described the general conformal framework and its transformations. However, many of those operators were not employed in the present work. Indeed, in the case that only rigid transformations are needed, we will limit ourselves to the use of the *Affine Plane*, which is an $(n + 1)$-dimensional subspace of the hyperplane of reference $\mathbb{P}(e_\infty, e_0)$.

We have chosen to work in the $\mathcal{G}_{4,1}$ algebra. Since we deal with homogeneous points, the particular choice of null vectors does not affect the properties of the conformal geometry. Thus, for this work, we choose to define these vectors as shown in Eqs. (2)–(6).

Points in the affine plane $\boldsymbol{x} \in \mathbb{R}^{4,1}$ are formed as follows:

$$\boldsymbol{x}^a = \mathbf{x}_e + e_0 \quad (35)$$

where $\mathbf{x}_e \in \mathbb{R}^3$. From this equation, we note that e represents the origin (by setting $\mathbf{x}_e = 0$); similarly, e_∞ represents the point at infinity. The previous equation, allows its

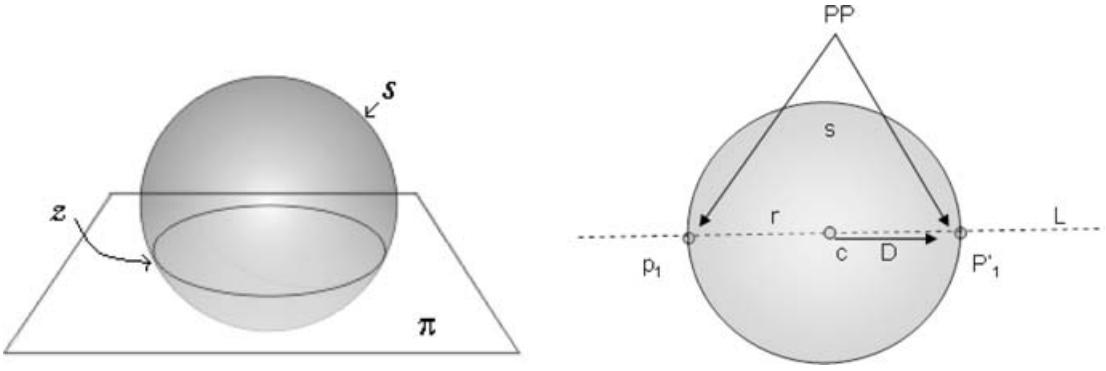

Fig. 3. (a) The meet of a sphere and a plane. (b) Pair of points resulting from the meet between a line and a sphere.

normalization, which is expressed as

$$e_\infty \cdot \boldsymbol{x}^a = 1. \tag{36}$$

In this framework, the conformal mapping equation is expressed with

$$\boldsymbol{x}_c = \mathbf{x}_e + \frac{1}{2}\mathbf{x}_e^2 e_\infty + e = \boldsymbol{x}^a - \mathbf{x}_e^2 e_\infty. \tag{37}$$

For the case when we will be exclusively working on the affine plane, we will be mainly concerned with a simplified version of the *rejection*. Noting that $\boldsymbol{E} = e_\infty \wedge e_0 = e_\infty \wedge e$, we write an equation for rejection asfollows:

$$\begin{aligned} P_E^\perp(\boldsymbol{x}_c) &= (\boldsymbol{x}_c \wedge \boldsymbol{E})\boldsymbol{E} = (\boldsymbol{x}_c \wedge \boldsymbol{E}) \cdot \boldsymbol{E} \\ &= (e_\infty \wedge e_0) \cdot e_0 + (\boldsymbol{x}_c \wedge e_\infty) \cdot e_0 \\ \mathbf{x}_e &= -e_0 + (\boldsymbol{x}_c \wedge e_\infty) \cdot e_0. \end{aligned} \tag{38}$$

Now, since the points in the affine plane have the form $\boldsymbol{x}_a = \boldsymbol{x}_e + e_0$, we conclude that

$$\boldsymbol{x}^a = (\boldsymbol{x}_c \wedge e_\infty) \cdot e_0 \tag{39}$$

is the mapping from the horosphere to the affine plane.

3.1. Lines and planes

The lines and planes in the affine plane are expressed in a similar manner to their conformal counterparts as the *join* of 2 and 3 points, respectively.

$$\boldsymbol{L}^a = \boldsymbol{x}_1^a \wedge \boldsymbol{x}_2^a \tag{40}$$

$$\Pi^a = \boldsymbol{x}_1^a \wedge \boldsymbol{x}_2^a \wedge \boldsymbol{x}_3^a. \tag{41}$$

Note that unlike their conformal counterparts, the line is a *bivector* and the plane is a *trivector*. As seen earlier, these equations produce a moment-direction representation

$$\boldsymbol{L}^a = e_\infty \mathbf{d} + \boldsymbol{B} \tag{42}$$

where $\mathbf{d}$ is a vector representing the direction of the line, and B is a bivector representing the (orthogonal) moment of the line. Similarly we have

$$\Pi^a = e_\infty \mathbf{n} + \delta e_{123} \tag{43}$$

where $\mathbf{n}$ is the normal vector to the plane and δ is a scalar representing the distance from the plane to the origin. Note that in any case, the direction and normal can be retrieved with $\mathbf{d} = e_\infty \cdot \boldsymbol{L}^a$ and $\mathbf{n} = e_\infty \cdot \Pi^a$, respectively.

In this framework, the intersection or *meet* has a simple expression too. Let $\boldsymbol{A}^a = a_1^a \wedge \cdots \wedge a_r^a$ and $\boldsymbol{B}^a = b_1^a \wedge \cdots \wedge b_s^a$, then the meetis defined as

$$\boldsymbol{A}^a \cap \boldsymbol{B}^a = \boldsymbol{A}^a \cdot (\boldsymbol{B}^a \cdot \bar{I}_{\boldsymbol{A}^a \cup \boldsymbol{B}^a}) \tag{44}$$

where $\bar{I}_{\boldsymbol{A}^a \cup \boldsymbol{B}^a}$ is either $e_{12}e_\infty$, $e_{23}e_\infty$, $e_{31}e_\infty$, or $e_{123}e_\infty$, according to which basis vectors span the largest common space of $\boldsymbol{A}^a$ and $\boldsymbol{B}^a$; see Section 2.

3.2. Directed distance

To derive our equations, we can use the line and the plane depicted in Fig. 4. We can see that any k-plane $\boldsymbol{A}^a$ consists of a momentum of degree k and a direction of degree $k-1$. Thus, if we take the inner product between the unit direction and the moment, we will gain a directed distance as the vector $\boldsymbol{p}^a$. If we dot the equation of $\boldsymbol{A}^a$ with $\bar{e}$ and divide this result by its norm, we get the unit direction $\boldsymbol{D}_u^a$

$$\boldsymbol{D}^a = \bar{e} \cdot \boldsymbol{A}_a \longrightarrow \boldsymbol{D}_u^a = \frac{\boldsymbol{D}_a^a}{\left|\boldsymbol{D}_a^a\right|} \tag{45}$$

which we can further use to compute a directed distance as follows:

$$\boldsymbol{p}^a = \boldsymbol{D}_u^a \cdot \boldsymbol{A}^a. \tag{46}$$

Here, the dot operation basically takes place between $\boldsymbol{D}_u^a$ and the moment part of $\boldsymbol{A}^a$. The point p^a is referred to the origin and it touches orthogonal the k-plane $\boldsymbol{A}^a$. Interesting enough,

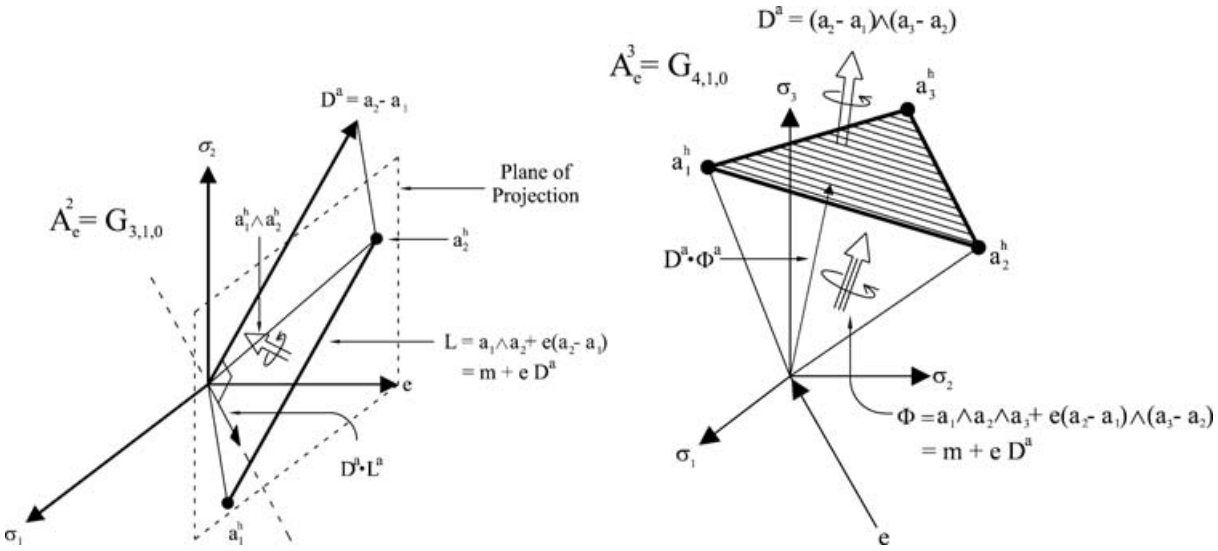

Fig. 4. (a) Line in the 2D affine space. (b) Plane in the 3D affine space (note that the 3D space is "lifted" by a null vector e.

the norm of $\boldsymbol{p}^a$ equals the Hesse distance. For the sake of simplicity in Figs. 4(a) and (b), only $\boldsymbol{D}^a \cdot \boldsymbol{L}^a$ and $\boldsymbol{D}^a \cdot \boldsymbol{\Phi}^a$ are shown, respectively.

Now, having this point on the first object, we can use it to compute the directed distance from the k-plane $\boldsymbol{A}^a$ parallel to the object $\boldsymbol{B}^a$ as follows:

$$d[\boldsymbol{A}^a, \boldsymbol{B}^a] = d[\boldsymbol{D}^a \cdot \boldsymbol{A}^a, \boldsymbol{B}^a] = d[(\bar{e} \cdot \boldsymbol{A}^a) \cdot \boldsymbol{A}^a, \boldsymbol{B}^a]. \quad (47)$$

4. Rigid Transformations

We can express rigid transformations in conformal geometry carrying out reflections between planes.

4.1. Reflection

In general the reflection operation can be used to build more complex conformal transformations. Here we will explain the geometry of a reflection. The reflection of a point x with respect to the plane π is equal to x minus twice the direct distance between the point and plane (see Fig. 5), that is, $x = x - 2(\pi \cdot x)\pi^{-1}$. To simplify this expression, we apply the property of the Clifford product of vectors, namely $2(b \cdot a) = ab + ba$. Then reflection could be written as

$$x' = x - (\pi x - x\pi)\pi^{-1} \quad (48)$$

$$x' = x - \pi x \pi^{-1} - x\pi\pi^{-1} \quad (49)$$

$$x' = -\pi x \pi^{-1}. \quad (50)$$

For any geometric entity Q, the reflection with respect to the plane π is given by

$$Q' = \pi Q \pi^{-1}. \quad (51)$$

4.2. Translation

The translation of conformal entities can be done by carrying out two reflections in parallel planes π_1 and π_2 (see Fig. 6), that is,

$$Q' = \underbrace{(\pi_2\pi_1)}_{T_a} Q \underbrace{(\pi_1^{-1}\pi_2^{-1})}_{\widetilde{T}_a} \quad (52)$$

$$T_a = (n + de_\infty)n = 1 + \frac{1}{2}ae_\infty = e^{-\frac{a}{2}e_\infty} \quad (53)$$

where $a = 2dn$.

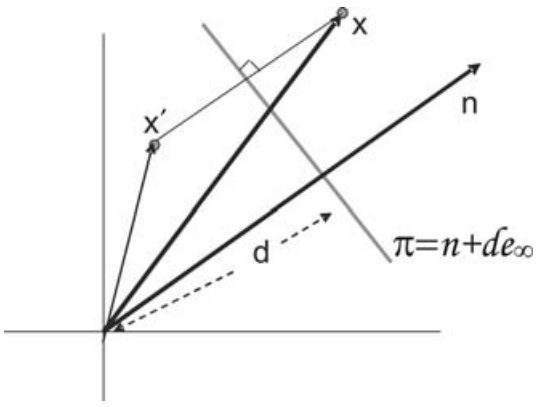

Fig. 5. Reflection of a point x with respect to the plane π.

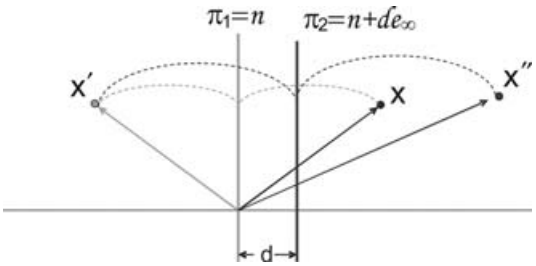

Fig. 6. Reflection about parallel planes.

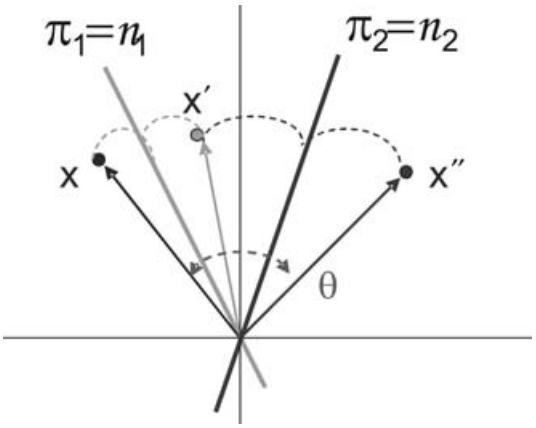

Fig. 7. Reflection about nonparallel planes.

4.3. Rotation

Rotation is the product of two reflections between nonparallel planes (see Fig. 7)

$$Q' = \underbrace{(\pi_2\pi_1)}_{R_\theta} Q \underbrace{(\pi_1^{-1}\pi_2^{-1})}_{\widetilde{R}_\theta} \quad (54)$$

or computing the conformal product of the normals of the planes

$$R_\theta = n_2 n_1 = \cos\left(\frac{\theta}{2}\right) - \sin\left(\frac{\theta}{2}\right) l = e^{-\frac{\theta}{2}l} \quad (55)$$

with $l = n_2 \wedge n_1$, and θ equal to twice the angle between the planes π_2 and π_1. The screw motion called *motor* related to an arbitrary axis L is $M = TR\widetilde{T}$

$$Q' = \underbrace{(TR\widetilde{T})}_{M_\theta} Q \underbrace{((T\widetilde{R}\widetilde{T}))}_{\widetilde{M}_\theta} \quad (56)$$

$$M_\theta = TR\widetilde{T} = \cos\left(\frac{\theta}{2}\right) - \sin\left(\frac{\theta}{2}\right) L = e^{-\frac{\theta}{2}L}. \quad (57)$$

4.4. Kinematic chains

The direct kinematics for serial robot arms is a succession of motors and it is valid for points, lines, planes, circles, and spheres

$$Q' = \prod_{i=1}^{n} M_i \, Q \prod_{i=1}^{n} \widetilde{M}_{n-i+1}. \quad (58)$$

5. Differential Kinematics

The direct kinematics Eq. (58) can be used for points as

$$x'_p = \prod_{i=1}^{n} M_i x_p \prod_{i=1}^{n} \widetilde{M}_{n-i+1}. \tag{59}$$

This equation could be used in conformal geometric algebra, using motors to represent 3D rigid transformations similar to the motor algebra.[5] Now we give an expression for differential kinematics through the total differentiation of Eq. (59) as follows:

$$\mathrm{d}x'_p = \sum_{j=1}^{n} \partial_{q_j} \left(\prod_{i=1}^{n} M_i x_p \prod_{i=1}^{n} \widetilde{M}_{n-i+1} \right) \mathrm{d}q_j. \tag{60}$$

Each term of the sum is the product of two functions in q_j; then the differential reads

$$\mathrm{d}x'_p = \sum_{j=1}^{n} \left[\partial_{q_j} \left(\prod_{i=1}^{j} M_i \right) \prod_{i=j+1}^{n} M_i x_p \prod_{i=1}^{n} \widetilde{M}_{n-i+1} + \prod_{i=1}^{n} M_i x_o \prod_{i=1}^{n-j} \widetilde{M}_{n-i+1} \partial_{q_j} \left(\prod_{i=n-j+1}^{n} \widetilde{M}_{n-i+1} \right) \right] \mathrm{d}q_j. \tag{61}$$

Since $M = e^{-\frac{1}{2}qL}$, the differential of the motor is $\mathrm{d}(M) = -\frac{1}{2} ML \mathrm{d}q$. Thus, we can write the partial differential of the motor's product as follows:

$$\partial_{q_j} \left(\prod_{i=1}^{j} M_i \right) = -\frac{1}{2} \prod_{i=1}^{j} M_i L_j = -\frac{1}{2} \left(\prod_{i=1}^{j-1} M_i \right) L_j M_j. \tag{62}$$

Similarly, the differential of the $\widetilde{M} = e^{\frac{1}{2}qL}$ give us $\mathrm{d}(\widetilde{M}) = \frac{1}{2} ML \mathrm{d}q$ and the differential of the product is

$$\partial_{q_j} \left(\prod_{i=n-j+1}^{n} \widetilde{M}_{n-i+1} \right) = \frac{1}{2} \widetilde{M}_j L_j \prod_{i=n-j+2}^{n} \widetilde{M}_{n-i+1}. \tag{63}$$

Replacing Eqs. (62) and (63) in Eq. (61), we get

$$\mathrm{d}x'_p = \sum_{j=1}^{n} \left[-\frac{1}{2} \prod_{i=1}^{j-1} M_i L_j M_j \prod_{i=j+1}^{n} M_i x_p \prod_{i=1}^{n} \widetilde{M}_{n-i+1} + \frac{1}{2} \prod_{i=1}^{n} M_i x_o \prod_{i=1}^{n-j} \widetilde{M}_{n-i+1} \widetilde{M}_j L_j \prod_{i=1}^{j-1} \widetilde{M}_{j-i} \right] \mathrm{d}q_j \tag{64}$$

which can be further simplified as

$$\mathrm{d}x'_p = -\frac{1}{2} \sum_{j=1}^{n} \left[\prod_{i=1}^{j-1} M_i \left(L_j \left(\prod_{i=j}^{n} M_i x_o \prod_{i=1}^{n-j+1} \widetilde{M}_{n-i+1} \right) - \left(\prod_{i=j}^{n} M_i x_p \prod_{i=1}^{n-j+1} \widetilde{M}_{n-i+1} \right) L_j \right) \prod_{i=1}^{j-1} \widetilde{M}_{j-i} \right] \mathrm{d}q_j. \tag{65}$$

Note that the product of a vector with an r-vector is given by

$$a \cdot B_r = \frac{1}{2} \left(aB_r + (-1)^{r+1} B_r a \right). \tag{66}$$

Using Eq. (66), we can simplify Eq. (65), since L is a bivector and x_p is a vector. Then, we can rewrite Eq. (65) as follows:

$$\mathrm{d}x'_p = \sum_{j=1}^{n} \left[\left(\prod_{i=1}^{j-1} M_i \right) \left(\left(\prod_{i=j}^{n} M_i x_p \prod_{i=1}^{n-j+1} \widetilde{M}_{n-i+1} \right) \cdot L_j \right) \times \left(\prod_{i=1}^{j-1} \widetilde{M}_{j-i} \right) \right] \mathrm{d}q_j. \tag{67}$$

Similar to the case of points, all the transformations in conformal geometric algebra can also be applied to the lines. Thus

$$\mathrm{d}x'_p = \sum_{j=1}^{n} \left[\left(\prod_{i=1}^{j-1} M_i \prod_{i=j}^{n} M_i x_p \prod_{i=1}^{n-j+1} \widetilde{M}_{n-i+1} \prod_{i=1}^{j-1} \widetilde{M}_{j-i} \right) \cdot \left(\prod_{i=1}^{j-1} M_i L_j \prod_{i=1}^{j-1} \widetilde{M}_{j-i} \right) \right] \mathrm{d}q_j. \tag{68}$$

Since $\prod_{i=1}^{j-1} M_i \prod_{i=j}^{n} M_i = \prod_{i=1}^{n} M_i$, we have

$$\mathrm{d}x'_p = \sum_{j=1}^{n} \left[\left(\prod_{i=1}^{n} M_i x_p \prod_{i=1}^{n} \widetilde{M}_{n-i+1} \right) \cdot \left(\prod_{i=1}^{j-1} M_i L_j \prod_{i=1}^{j-1} \widetilde{M}_{j-i} \right) \right] \mathrm{d}q_j. \tag{69}$$

Recall Eq. (59) of the direct kinematics, since in Eq. (69) x'_p appears again. We can replace Eq. (59) in Eq. (69) to get

$$\mathrm{d}x'_p = \sum_{j=1}^{n} \left[x'_p \cdot \left(\prod_{i=1}^{j-1} M_i L_j \prod_{i=1}^{j-1} \widetilde{M}_{j-i} \right) \right] \mathrm{d}q_j. \tag{70}$$

If we define L' as function of L as follows:

$$L'_j = \prod_{i=1}^{j-1} M_i L_j \prod_{i=1}^{j-1} \widetilde{M}_{j-i} \tag{71}$$

we get a very compact expression of differential kinematics

$$\mathrm{d}x'_p = \sum_{j=1}^{n} [x'_p \cdot L'_j]\mathrm{d}q_j. \tag{72}$$

In this way we can finally write

$$\dot{x}'_p = (x'_p \cdot L'_1 \ \cdots \ x'_p \cdot L'_n) \begin{pmatrix} \dot{q}_1 \\ \vdots \\ \dot{q}_n \end{pmatrix}. \tag{73}$$

After we have given the theoretical background of our geometric approach for modeling and controlling robotic devices, we will now present inverse kinematics computations of robot devices, and also some interesting applications of these geometric techniques using real images captured by a mobile robot.

6. Inverse Kinematics of Robot Devices

This section presents a new way to compute inverse kinematics using the conformal geometric algebra. We illustrate the computations of the inverse kinematics of a robot arm of five DOF and that of a binocular head of a robot. The striking aspect of our approach is the use of a mathematical language of spheres.

6.1. Inverse kinematics of robot arms

The inverse kinematics problem consists of determining the angles of each joint of the robot arm $\theta_1 \cdots \theta_5$, so that the end-effector reaches a specific point (p_t), the plane of the gripper is parallel to the plane π_t, and its direction is l_t (see Fig. 8). We show how to find the values of $\theta_1 \cdots \theta_5$ using the conformal approach. For this purpose, we use geometric entities like circles, planes, and spheres. For its solution, the problem will be divided in five steps.

Step 1: Find the position of the point p_2.

The sphere with center at p_t and radius d_3 is given by

$$S_t = p_t - \frac{1}{2} d_3^2 e_\infty. \tag{74}$$

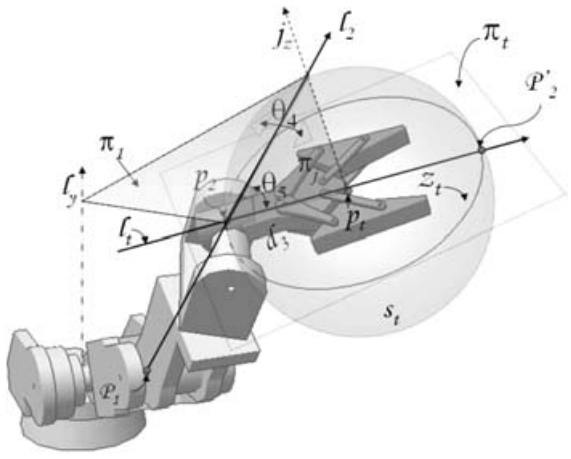

Fig. 8. Point of touch p_t, plane of grasp π_t, and direction l_t.

Now we consider the second constraint that forces the gripper to be parallel to the plane π_t. This constraint reduces the possible positions of p_2, forcing this point to lie on the circle z_t, which is the intersection of the plane π_t with the sphere S_t (see Fig. 8).

$$z_t = S_t \wedge \pi_t. \tag{75}$$

Finally, we consider the last alignment condition. The tangent to the circle z_t at the point p_2 is orthogonal to the line l_t; therefore, l_t passes through the point p_t, it belongs to the plane π_t and it intersects the circle z_t at point p_2.

$$P_{p2} = l_t \wedge S_t. \tag{76}$$

Step 2: Compute the position of the point p_0.

The y-axis (l_y) is the line going through the origin with direction e_2 (see Eq. (27))

$$l_y^* = e_2 E. \tag{77}$$

When the base rotates around the y-axis (see Fig. 9), the point p_0 describes the circle z_0. This circle is the intersection of the plane π_0 and the sphere with center at the origin and radius d_0.

$$S_0 = e_o - \frac{d_o^2}{2} e_\infty \tag{78}$$

$$\pi_0 = e_2 + h e_\infty \tag{79}$$

$$z_0 = S_o \wedge \pi_0. \tag{80}$$

We have restricted the position of the point p_0 to lie on the circle z_0, but there is another restriction: point p_0 must lie on the plane π_1 generated by the y-axis (l_y; Eq. (77)) and the point p_2 calculated in step 1 (Eq. (76)), as we can see in Fig. 8, so that p_0 can be determined by intersecting the plane π_1 with the circle z_0

$$\pi_1^* = l_y^* \wedge p_2 \tag{81}$$

$$P_{p0} = z_0 \wedge \pi_1. \tag{82}$$

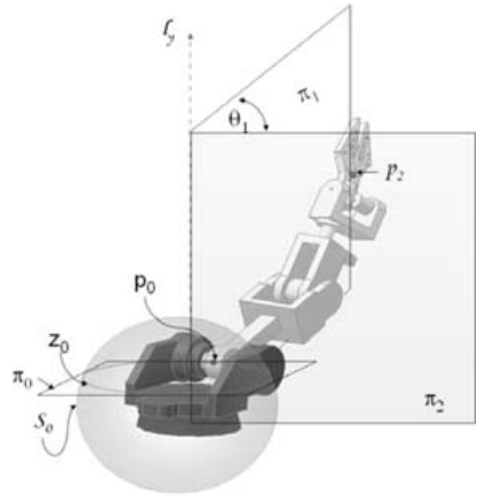

Fig. 9. Point p_0 as an intersection of the planes π_1, π_0, and the sphere s_0.

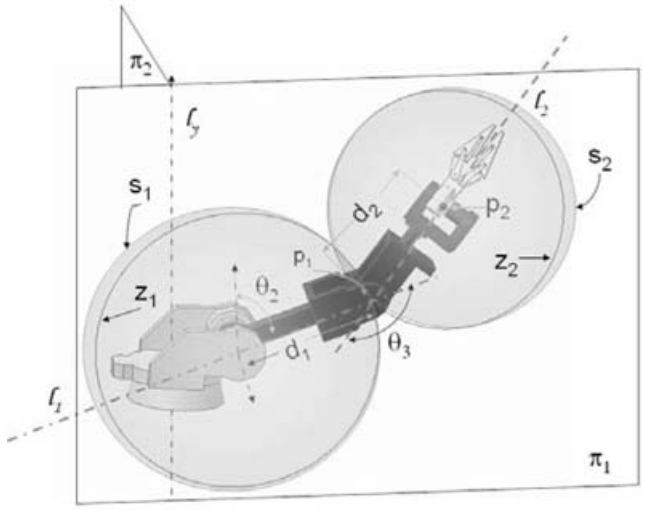

Fig. 10. Point p_1 as an intersection of the spheres s_1, s_2, and the plane π_1.

We have a pair of points as solution and we choose one of them.

Step 3: Determine the position of the point p_1.

Computing the point p_1 is usually a difficult task because it is the intersection of two circles; however, using conformal geometry we can easily determine this point by intersecting the spheres S_1 and S_2 with the plane π_1, as can be seen in Fig. 10.

$$S_1 = p_1 - \frac{d_1^2}{2} e_\infty \tag{83}$$

$$S_2 = p_2 - \frac{d_2^2}{2} e_\infty \tag{84}$$

$$P_{p1} = S_1 \wedge S_2 \wedge \pi_1. \tag{85}$$

Similar to the previous step, we choose one of the two points.

Step 4: Determine the lines and planes between the joints of the robot.

Once p_0, p_1, and p_2 have been determined, the lines l_1, l_2, and l_3, and the planes π_2 and π_3 can be defined. These lines and planes will be used to calculate the angles $\theta_1 \cdots \theta_5$.

$$\pi_3^* = p_1 \wedge p_2 \wedge p_t \wedge e_\infty \tag{86}$$

$$\pi_2^* = e_3 I_c \tag{87}$$

$$l_1^* = p_0 \wedge p_1 \wedge e_\infty \tag{88}$$

$$l_2^* = p_1 \wedge p_2 \wedge e_\infty \tag{89}$$

$$l_3^* = p_2 \wedge p_3 \wedge e_\infty. \tag{90}$$

Step 5: Find the angles $\theta_1 \cdots \theta_5$.

Once we have all the geometric entities, the computation of the angles is a trivial step

$$\cos(\theta_1) = \frac{\pi_1^* \cdot \pi_2^*}{|\pi_1^*||\pi_2^*|} \tag{91}$$

$$\cos(\theta_2) = \frac{l_1^* \cdot l_y^*}{|l_1^*||l_y^*|} \tag{92}$$

$$\cos(\theta_3) = \frac{l_1^* \cdot l_2^*}{|l_1^*||l_2^*|} \tag{93}$$

$$\cos(\theta_4) = \frac{\pi_1^* \cdot \pi_3^*}{|\pi_1^*||\pi_3^*|} \tag{94}$$

$$\cos(\theta_5) = \frac{l_2^* \cdot l_3^*}{|l_2^*||l_3^*|}. \tag{95}$$

Since the robot arm has just five DOF, the line l_2 should intersect the line j_z. Therefore, it is not always possible to satisfy the constrains π_t, p_t, and l_t. But we can solve for π_t and p_t as follows.

There is an infinite number of solutions to reach the point p_t because p_2 can remain at a distance of d_3 around p_t (see Fig. 11). That is, the point p_2 lies on the sphere S_t with center at p_t and radius d_3

$$S_t = p_t - \frac{1}{2} d_3^2 e_\infty. \tag{96}$$

However, we must also satisfy the condition that the plane of the gripper should be parallel to the plane π_t. Hence, p_2 must lie on the circle z_t, which is the intersection of the sphere S_t (Eq. (96)) and the plane π_t

$$z_t = S_t \wedge \pi_t. \tag{97}$$

Since the robotic arm has only five DOFs, the point p_2 can only be in two positions, namely the points on the circle z_t that are closest to, and farthest from l_y. Finding these points is easy because they are the intersections of the plane π_j and

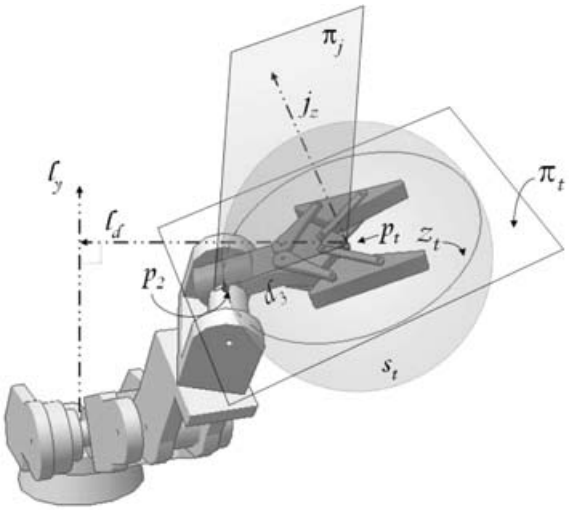

Fig. 11. Point p_2 given by the intersection of the plane π_t, s_t, and the plane π_j.

the circle z_t

$$j_z^* = z_t \wedge e_\infty \tag{98}$$

$$l_d = d(p_t, l_y^*) \wedge p_t \wedge e_\infty \tag{99}$$

$$\pi_j^* = j_z^* \wedge (l_d^* E) \tag{100}$$

$$P_{p2} = \pi_j \wedge z_t. \tag{101}$$

In this way, we have two points of solution. However, we need to choose one. We choose the point that is mechanically possible to reach and follow the steps 2–5 for computing the angles.

7. Visual Jacobian

This section includes the Jacobian visual of the camera useful for closing the loop between the visual 3D space and the 3D mechanical world. The rows of the camera projection matrix are interpreted as optical planes (π_1, π_2, π_3). Each one of these optical planes is written in conformal geometry as a vector. Then, the camera in general position is described by the collection of three conformal planes

$$P = \begin{pmatrix} \pi_1 \\ \pi_2 \\ \pi_3 \end{pmatrix}. \tag{102}$$

A conformal point (that represents a point in 3D visual space) is projected to the camera projection matrix by computing the inner product between this point and each one of the camera's planes. The same is true for its derivative

$$m = \begin{pmatrix} \pi_1 \cdot X \\ \pi_2 \cdot X \\ \pi_3 \cdot X \end{pmatrix} \qquad \dot{m} = \begin{pmatrix} \pi_1 \cdot \dot{X} \\ \pi_2 \cdot \dot{X} \\ \pi_3 \cdot \dot{X} \end{pmatrix}. \tag{103}$$

The point in the image is computed as follows:

$$s = \begin{pmatrix} \frac{\pi_1 \cdot X}{\pi_3 \cdot X} \\ \frac{\pi_2 \cdot X}{\pi_3 \cdot X} \end{pmatrix} \tag{104}$$

and its derivative

$$\dot{s} = \begin{pmatrix} \dot{m}_1 \frac{1}{m_3} + m_1 \left(\frac{-\dot{m}_3}{m_3^2} \right) \\ \dot{m}_2 \frac{1}{m_3} + m_2 \left(\frac{-\dot{m}_3}{m_3^2} \right) \end{pmatrix} \tag{105}$$

simplifying

$$\dot{s} = \kappa \begin{pmatrix} m_3 \dot{m}_1 - m_1 \dot{m}_3 \\ m_3 \dot{m}_2 - m_2 \dot{m}_3 \end{pmatrix} \tag{106}$$

where $\kappa = \frac{1}{m_3^2}$. Evaluating the values of m and $\dot{m}$ in Eq. (106), we get

$$\dot{s} = \kappa \begin{pmatrix} [(\pi_3 \cdot X)\pi_1 - (\pi_1 \cdot X)\pi_3] \cdot \dot{X} \\ [(\pi_3 \cdot X)\pi_2 - (\pi_2 \cdot X)\pi_3] \cdot \dot{X} \end{pmatrix}. \tag{107}$$

This equation can be reduced using Eq. (1.10) of the Hestenes Chapter in ref. 15

$$\dot{s} = \kappa X \cdot \begin{pmatrix} \pi_1 \wedge \pi_3 \\ \pi_2 \wedge \pi_3 \end{pmatrix} \cdot \dot{X}. \tag{108}$$

The wedge product of two planes equals the representation of the line of their intersection. In this paper, L_x and L_y are used to denote the intersection of the planes (π_1, π_3) and (π_2, π_3), respectively.

$$L_x := \pi_1 \wedge \pi_3 \tag{109}$$

$$L_y := \pi_2 \wedge \pi_3. \tag{110}$$

Using these definitions, Eq. (108) can be rewritten as

$$\dot{s} = \kappa X \cdot \begin{pmatrix} L_x \\ L_y \end{pmatrix} \cdot \dot{X}. \tag{111}$$

In order to close the loop of perception and action, we compute the relationship between visual velocities in the image of the camera and joint velocities in the mechanical structure.

Remembering the equation of the differential kinematics for a pan–tilt unit, which relates joint and end-effector velocities, we can claim

$$\dot{X}' = (X' \cdot L_1' \; X' \cdot L_2') \begin{pmatrix} \dot{q}_1 \\ \dot{q}_2 \end{pmatrix}. \tag{112}$$

Combining Eqs. (108) and (112), a new equation is obtained that relates the joint and image velocities

$$\dot{s} = \kappa \begin{pmatrix} (X' \cdot L_x') \cdot (X' \cdot L_1') & (X' \cdot L_x') \cdot (X' \cdot L_2') \\ (X' \cdot L_y') \cdot (X' \cdot L_1') & (X' \cdot L_y') \cdot (X' \cdot L_2') \end{pmatrix} \dot{q}. \tag{113}$$

This expression is called the visual Jacobian and it can be used to implement a robust control law for visual tracking using a pan–tilt unit.

7.1. Inverse kinematics for a pan–tilt unit

The problem involves determining the angles θ_{tilt} and θ_{pan} of a stereo-head, so that the cameras fix at the point p_t. We will now show how we can find the values of θ_{pan} and θ_{tilt} using the conformal approach. For its solution, The problem will be divided in three steps.

Step 1: Determine the point p_2.

When the θ_{tilt} rotates and the bases rotate (θ_{pan}) around the l_y (see Fig. 12), the point p_2 describes a sphere s_1. This sphere has center at the point p_1 and has a radius d_2.

$$S_1 = p_1 - \frac{d_2^2}{2} e_\infty. \tag{114}$$

Also the point p_t can be locked from every point around it. That is, the point p_2 is in the sphere

$$S_2 = p_t - \frac{d_3^2}{2} e_\infty \tag{115}$$

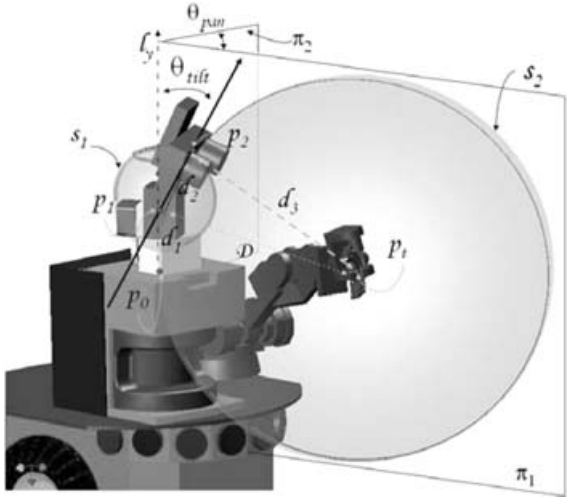

Fig. 12. Point p_2 given by the intersection of the plane π_1 and the spheres s_1 and s_2.

where d_3 is the distance between point p_t and the cameras, and we can calculate d_3 using the Pythagorean theorem $d_3^2 = D^2 - d_2^2$, where D is the direct distance between p_t and p_1. We have restricted the position of the point p_2, but there is another restriction: the vector going from the point p_2 to the point p_t must lie on the plane π_1 generated by the l_y-axis ($l_y^* = p_0 \wedge p_1 \wedge e_\infty$) and the point p_t, as we can see in Fig. 12. Hence, p_2 can be determined by intersecting the plane π_1 with the spheres s_1 and s_2 as follows:

$$\pi_1^* = l_y^* \wedge p_t, \qquad P_{p2} = s_1 \wedge \pi_1 \wedge s_2. \tag{116}$$

Step 2: Determine the lines and planes.

Once p_2 have been determined, the line l_2 and the plane π_2 can be defined. This line and plane will be useful to calculate the angles θ_{tilt} and θ_{pan}.

$$l_2^* = p_1 \wedge p_2 \wedge e_\infty, \qquad \pi_2^* = l_y^* \wedge e_3. \tag{117}$$

Step 3: Find the angles θ_{tilt} and θ_{pan}.

Once we have all the geometric entities, the computation of the angles is a trivial step.

$$\cos(\theta_{\text{pan}}) = \frac{\pi_1^* \cdot \pi_2^*}{|\pi_1^*||\pi_2^*|}, \qquad \cos(\theta_{\text{tilt}}) = \frac{l_1^* \cdot l_y^*}{|l_1^*||l_y^*|}. \tag{118}$$

8. Following Geometric Primitives for Object Manipulation

In this section, we will show how to perform certain object manipulation tasks now in the context of conformal geometric algebra. First, we will solve the problem of positioning the gripper of the arm in a certain position of space disregarding the grasping plane or the gripper's alignment. Then, we will illustrate how the robotic arm can follow linear paths.

8.1. Touching a point

In order to reconstruct the point of interest, we back-project two rays extending from two views of a given scene (see Fig. 13). These rays will not intersect in general, due to noise. Hence, we compute the directed distance between these lines and use the middle point as target. Once the 3D point p_t is computed with respect to the cameras' framework, we transform it to the arm's coordinate system.

Once we have a target point with respect to the arm's framework, there are three cases to consider. There might be several solutions [Figs. 14(a) and 15(a)], a single solution [Fig. 14(b)], or the point may be impossible to reach [Fig. 15(b)].

In order to distinguish between these cases, we create a sphere $\boldsymbol{S}_t = \boldsymbol{p}_t - \frac{1}{2}d_3^2 e_\infty$ centered at the point $\boldsymbol{p}_t$, and intersect it with the bounding sphere $\boldsymbol{S}_e = \boldsymbol{p}_0 - \frac{1}{2}(d_1 + d_2)^2 e_\infty$ of the other joints [Fig. 14(a) and (b)], producing the circle $\boldsymbol{z}_s = \boldsymbol{S}_e \wedge \boldsymbol{S}_t$.

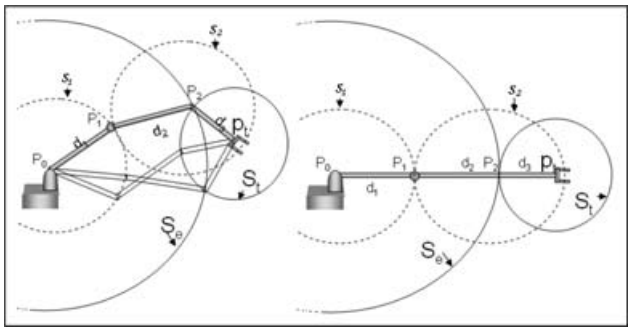

Fig. 14. (a) S_e and S_t meet (infinite solutions). (b) S_e and S_t are tangents (single solution).

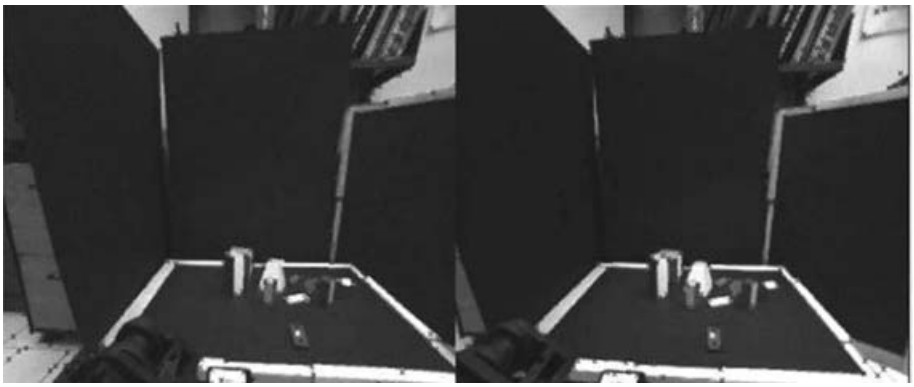

Fig. 13. Point of interest in both the cameras (p_t).

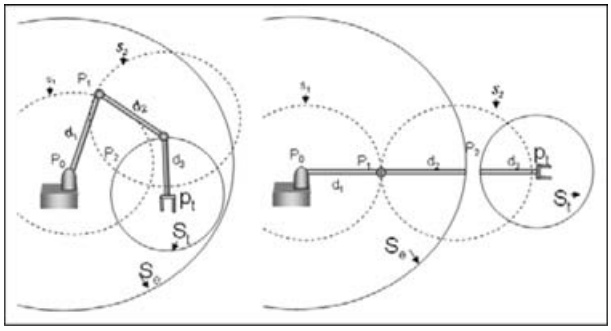

Fig. 15. (a) S_t inside S_e produces infinite solutions. (b) S_t outside S_e, produces no solution.

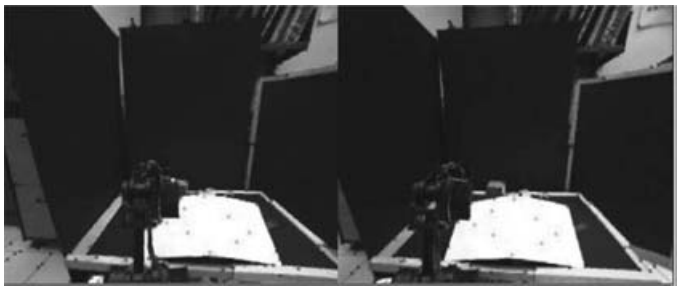

Fig. 17. Images acquired by the binocular system of the robot "Geometer" showing the points on each plane.

If the spheres $\boldsymbol{S}_t$ and $\boldsymbol{S}_e$ intersect, then we have a solution circle z_s that represents all the possible positions point $\boldsymbol{p}_2$ (Fig. 14) may have in order to reach the target. If the spheres are tangent, then there is only one point of intersection and a single solution to the problem, as shown in Fig. 14(b).

If the spheres do not intersect, then there are two possibilities. The first case is that $\boldsymbol{S}_t$ is outside the sphere $\boldsymbol{S}_e$. In this case, there is no solution since the arm cannot reach the point $\boldsymbol{p}_t$, as shown in Fig 15(b). On the other hand, if the sphere $\boldsymbol{S}_t$ is inside $\boldsymbol{S}_e$, then we have a sphere of solutions. In other words, we can place the point $\boldsymbol{p}_2$ anywhere inside $\boldsymbol{S}_t$, as shown in Fig. 15(a). For this case, we arbitrarily choose the upper point of the sphere $\boldsymbol{S}_t$.

In the experiment shown in Fig. 16(a), the sphere $\boldsymbol{S}_t$ is placed inside the bounding sphere $\boldsymbol{S}_e$. Therefore, the point selected by the algorithm is the upper limit of the sphere, as shown in Fig. 16(a) and (b). The last joint is completely vertical.

8.2. Line of intersection of two planes

In the industry, mainly in the sector dedicated to car assembly, it is often required to weld pieces. However, due to several factors, these pieces are not always in the same position complicating this task and making this process almost impossible to automate. In many cases, the requirement is to weld pieces of straight lines when no points on the line are available. This is the problem that has been dealt with in the following experiment.

If we do not have points on the line of interest, then we find this line via the intersection of two planes (the welding planes). In order to determine each plane, we need three points. The 3D coordinates of the points are triangulated using the stereo-vision system of the robot yielding a configuration like the one shown in Fig. 17.

Once the 3D coordinates of the points in space have been computed, we can find each plane with $\pi^* = \boldsymbol{x}_1 \wedge \boldsymbol{x}_2 \wedge \boldsymbol{x}_3 \wedge e_\infty$ and $\pi'^* = \boldsymbol{x}'_1 \wedge \boldsymbol{x}'_2 \wedge \boldsymbol{x}'_3 \wedge e'_\infty$. The line of intersection is computed via the *meet* operator $\boldsymbol{l} = \pi' \cap \pi$. In Fig. 18(a), we show a simulation of the arm following the line produced by the intersection of these two planes.

Once the line of intersection $\boldsymbol{l}$ is computed, it suffices with translating it on the plane $\psi = \boldsymbol{l}^* \wedge e_2$ [Fig. 18(b)] using the translator $\boldsymbol{T}_1 = 1 + \gamma e_2 e_\infty$, in the direction of e_2 (the y-axis) a distance γ. Furthermore, we build the translator $\boldsymbol{T}_2 = 1 + d_3 e_2 e_\infty$ with the same direction (e_2), but with a separation d_3 that corresponds to the size of the gripper. Once the translators have been computed, we find the lines $\boldsymbol{l}'$ and $\boldsymbol{l}''$ by translating the line $\boldsymbol{l}$ with $\boldsymbol{l}' = \boldsymbol{T}_1 \boldsymbol{l} \boldsymbol{T}_1^{-1}$, and $\boldsymbol{l}'' = \boldsymbol{T}_2 \boldsymbol{l}' \boldsymbol{T}_2^{-1}$.

The next step after computing the lines, is to find the points $\boldsymbol{p}_t$ and $\boldsymbol{p}_2$ that represent the places where the arm will start and finish its motion, respectively. These points were given manually, but they may be computed with the intersection of the lines $\boldsymbol{l}'$ and $\boldsymbol{l}''$ with a plane that defines the desired depth. In order to make the motion over the line, we build a translator $\boldsymbol{T}_L = 1 - \Delta_L \boldsymbol{l} e_\infty$ with the same direction as l, as shown in Fig. 18(b). Then, this translator is applied to the points $\boldsymbol{p}_2 = \boldsymbol{T}_L \boldsymbol{p}_2 \boldsymbol{T}_L^{-1}$ and $\boldsymbol{p}_t = \boldsymbol{T}_L \boldsymbol{p}_t \boldsymbol{T}_L^{-1}$ in an iterative fashion to yield a displacement Δ_L on the robotic arm.

By placing the endpoint over the lines and p_2 over the translated line, and by following the path with a translator in the direction of l, we get a motion over l as seen in the image sequence of Fig. 19.

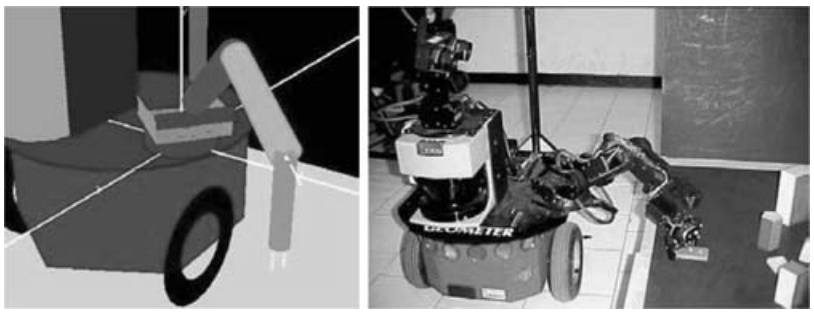

Fig. 16. (a) Simulation of the robotic arm touching a point. (b) Robot "Geometer" touching a point with its arm.

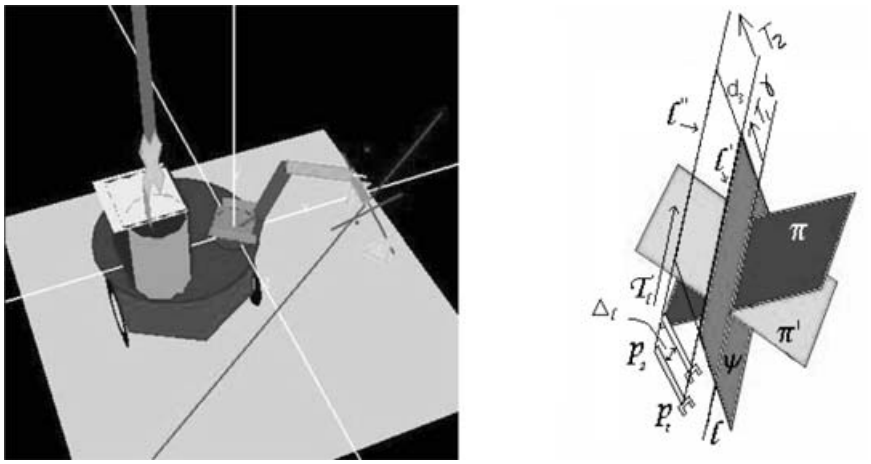

Fig. 18. (a) Simulation of the arm following the path of a line produced by the intersection of the two planes. (b) Guiding lines for the robotic arm produced by the intersection (meet) of planes and vertical translation.

Fig. 19. Image sequence of a linear-path motion.

8.3. Following a spherical path

This experiment involves following the path of a spherical object at a certain fixed distance from it. For this experiment, only four points on the object are available [Fig. 20(a)].

After acquiring the four 3D points, we compute the sphere $\boldsymbol{S}^* = \boldsymbol{x}_1 \wedge \boldsymbol{x}_2 \wedge \boldsymbol{x}_3 \wedge \boldsymbol{x}_4$. In order to place the point $\boldsymbol{p}_2$ in such a way that the arm points toward the sphere, the sphere was expanded using two different dilators. This produces a sphere that contains $\boldsymbol{S}^*$ and ensures that a fixed distance between the arm and $\boldsymbol{S}^*$ is preserved, as shown in Fig. 20(b).

The dilators are computed as follows:

$$\boldsymbol{D}_\gamma = e^{-\frac{1}{2}\ln(\frac{\gamma+\rho}{\rho})\boldsymbol{E}} \tag{119}$$

$$\boldsymbol{D}_d = e^{-\frac{1}{2}\ln(\frac{d_3+\gamma+\rho}{\rho})\boldsymbol{E}}. \tag{120}$$

The spheres $\boldsymbol{S}_1$ and $\boldsymbol{S}_2$ are computed by dilating $\boldsymbol{S}_t$

$$\boldsymbol{S}_1 = \boldsymbol{D}_\gamma \boldsymbol{S}_t \boldsymbol{D}_\gamma^{-1} \tag{121}$$

$$\boldsymbol{S}_2 = \boldsymbol{D}_d \boldsymbol{S}_t \boldsymbol{D}_d^{-1}. \tag{122}$$

We decompose each sphere in its parametric form as

$$\boldsymbol{p}_t = \boldsymbol{M}_1(\varphi)\boldsymbol{M}_1(\phi)\boldsymbol{p}_{s_1}\boldsymbol{M}_1^{-1}(\phi)\boldsymbol{M}_1^{-1}(\varphi) \tag{123}$$

$$\boldsymbol{p}_2 = \boldsymbol{M}_2(\varphi)\boldsymbol{M}_2(\phi)\boldsymbol{p}_{s_2}\boldsymbol{M}_2^{-1}(\phi)\boldsymbol{M}_2^{-1}(\varphi) \tag{124}$$

where p_s is any point on the sphere. In order to simplify the problem, we select the upper point on the sphere. To perform motion on the sphere, we vary the parameters φ and ϕ, and compute the corresponding $\boldsymbol{p}_t$ and $\boldsymbol{p}_2$ using Eqs. (123) and (124). The results of the simulation are shown in Fig. 21(a),

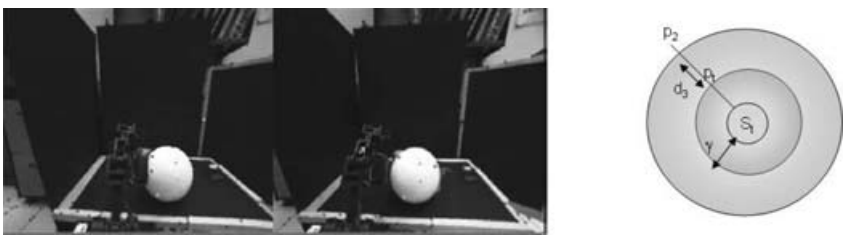

Fig. 20. (a) Points over the sphere as seen by the robot "Geometer". (b) Guiding spheres for the arm's motion.

Fig. 21. (a) Simulation of the motion over a sphere. (b) and (c) Two of the images in the sequence of the real experiment.

whereas the results of the real experiment can be seen in Fig. 21(b) and (c).

9. Differential Kinematic Control of a Pan–Tilt Unit

We will show an example using our new formulation of the Jacobian. This is the control of a pan–tilt unit where a stereo-vision system is fastened. This system is actually the binocular head of our mobile robot.

9.1. The pan–tilt unit

We implement algorithm for the velocity control of a pan–tilt unit (PTU; Fig. 22) assuming three DOFs. We consider the stereo-depth as one virtual DOF; thus, the PTU has a similar kinematic behavior as a robot with three DOFs.

In order to carry out a velocity control, we first need to compute the direct kinematics; this is very easy to do, as we know the axis lines

$$L_1 = -e_{31} \tag{125}$$

$$L_2 = e_{12} + d_1 e_1 e_\infty \tag{126}$$

$$L_3 = e_1 e_\infty. \tag{127}$$

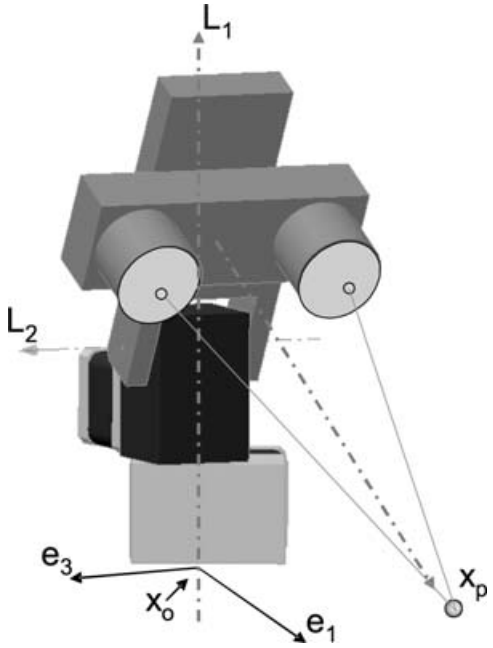

Fig. 22. Pan–tilt unit has two DOFs and the depth has a virtual DOF.

Since $M_i = e^{-\frac{1}{2}q_i L_i}$ and $\widetilde{M}_i = e^{\frac{1}{2}q_i L_i}$, we can compute the position of the end-effector using Eq. (59) as

$$x_p(q) = x'_p = M_1 M_2 M_3 x_p \widetilde{M}_3 \widetilde{M}_2 \widetilde{M}_1. \tag{128}$$

The estate variable representation of the system is as follows:

$$\begin{cases} \dot{x}'_p = x' \cdot (L'_1 \quad L'_2 \quad L'_3) \begin{pmatrix} u_1 \\ u_2 \\ u_3 \end{pmatrix} \\ y = x'_p \end{cases} \tag{129}$$

where the position of the end-effector at home position x_p is the conformal mapping of $x_{p_e} = d_3 e_1 + (d_1 + d_2) e_2$ (see Eq. 9), the line L'_i is the current position of L_i, and u_i is the velocity of the i-junction of the system. As L_3 is an axis at infinity, M_3 is a translator, that is, the virtual component is a prismatic junction.

9.2. Exact linearization via feedback

Now the following state feedback control law is chosen in order to get a new linear and controllable system.

$$\begin{pmatrix} u_1 \\ u_2 \\ u_3 \end{pmatrix} = (x'_p \cdot L'_1 \quad x'_p \cdot L'_2 \quad x'_p \cdot L'_3)^{-1} \begin{pmatrix} v_1 \\ v_2 \\ v_3 \end{pmatrix} \tag{130}$$

where $V = (v_1, v_2, v_3)^T$ is the new input to the linear system, then we rewrite the equations of the system

$$\begin{cases} \dot{x}'_p = V \\ y = x'_p. \end{cases} \tag{131}$$

9.3. Asymptotic output tracking

The problem of following a constant reference x_t is solved by computing the error between the end-effector position x'_p and the target position x_t as $e_r = (x'_p \wedge x_t) \cdot e_\infty$. The control law is then given by

$$V = -ke. \tag{132}$$

This error is small if the control system is doing its job. It is mapped to an error in the joint space using the inverse Jacobian.

$$U = J^{-1} V. \tag{133}$$

Doing the Jacobian $J = x'_p \cdot (L'_1\ L'_2\ L'_3)$

$$j_1 = x'_p \cdot (L_1) \tag{134}$$

$$j_2 = x'_p \cdot (M_1 L_2 \widetilde{M}_1) \tag{135}$$

$$j_3 = x'_p \cdot (M_1 M_2 L_3 \widetilde{M}_2 \widetilde{M}_1). \tag{136}$$

Once we have the Jacobian, it is easy to compute $\mathrm{d}q_i$ using the Crammer's rule.

$$\begin{pmatrix} u_1 \\ u_2 \\ u_3 \end{pmatrix} = (j_1 \wedge j_2 \wedge j_3)^{-1} \cdot \begin{pmatrix} V \wedge j_2 \wedge j_3 \\ j_1 \wedge V \wedge j_3 \\ j_1 \wedge j_2 \wedge V \end{pmatrix}. \tag{137}$$

This is possible because $j_1 \wedge j_2 \wedge j_3 = \det(J) I_e$. Finally, we have $\mathrm{d}q_i$ that will tend to reduce these errors. Due to the fact that the Jacobian has singularities, we should use the pseudo-inverse of Jacobian.

9.4. Pseudo-inverse of Jacobian

To avoid singularities, we compute the pseudo-inverse of the Jacobian matrix

$$J = [j_1 \quad j_2]. \tag{138}$$

Using the pseudo-inverse of Moore–Penrose

$$J^+ = (J^T J)^{-1} J^T. \tag{139}$$

Now evaluating J in Eq. (139)

$$J^+ = \frac{1}{\det(J^T J)} \begin{pmatrix} (j_2 \cdot j_2) j_1 - (j_2 \cdot j_1) j_2 \\ (j_1 \cdot j_1) j_2 - (j_2 \cdot j_1) j_1 \end{pmatrix}. \tag{140}$$

And using Clifford algebra we could simplify this equation further

$$\det(J^T J) = (j_1 \cdot j_1)(j_2 \cdot j_2) - (j_1 \cdot j_2)^2 \tag{141}$$

$$= (|j_1||j_2|)^2 - (|j_1||j_2|)^2 \cos^2(\theta), \tag{142}$$

$$= (|j_1||j_2|)^2 \sin^2(\theta), \tag{143}$$

$$= |j_1 \wedge j_2|^2 \tag{144}$$

calling θ the angle between vectors. Each row of J^+ could be simplified as follows:

$$(j_2 \cdot j_2) j_1 - (j_2 \cdot j_1) j_2 = j_2 \cdot (j_2 \wedge j_1) \tag{145}$$

$$(j_1 \cdot j_1) j_2 - (j_2 \cdot j_1) j_1 = j_1 \cdot (j_1 \wedge j_2). \tag{146}$$

Now Eq. (139) can be rewritten as

$$J^+ = \frac{1}{|j_1 \wedge j_2|^2} \begin{pmatrix} j_2 \cdot (j_2 \wedge j_1) \\ j_1 \cdot (j_1 \wedge j_2) \end{pmatrix} = \begin{pmatrix} j_2 \cdot (j_2 \wedge j_1)^{-1} \\ j_1 \cdot (j_1 \wedge j_2)^{-1} \end{pmatrix}. \tag{147}$$

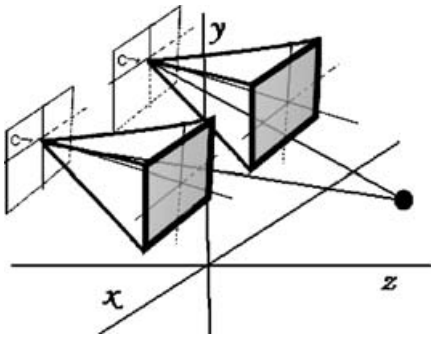

Fig. 23. Stereo cameras.

Using this equation, we can compute the input as $U = J^+ V$ that is equal to

$$U = (j_1 \wedge j_2)^{-1} \cdot \begin{pmatrix} V \wedge j_2 \\ j_1 \wedge V \end{pmatrix}. \tag{148}$$

9.5. Visual tracking

The target point is measured using two calibrated cameras (Fig. 23). With each image, we estimate the center of mass of the object in movement in order to do a retroprojection and finally estimate the 3D point. To compute the mass center, first we subtract the current image I_a to an image in memory I_b. The image in memory is the average of the last N images, and this help us to substract the background.

$$I_k(t) = I_a(t) - I_b(t-1) \times N \tag{149}$$

$$I_b(t) = (I_b(t-1) \times N + I_a)/(N+1). \tag{150}$$

After that the moments of x and y are computed. They are divided by the mass (pixels in movement), which corresponds to the intensity difference between the current and the memory images. In this way, the mass center is obtained.

$$x_o = \frac{\int_0^n \int_0^m I_k y \,\mathrm{d}x\,\mathrm{d}y}{\int_0^n \int_0^m I_k \,\mathrm{d}x\,\mathrm{d}y} \tag{151}$$

$$y_o = \frac{\int_0^n \int_0^m I_k x \,\mathrm{d}x\,\mathrm{d}y}{\int_0^n \int_0^m I_k \,\mathrm{d}x\,\mathrm{d}y}. \tag{152}$$

When the camera moves, the background changes and it is necessary to reset N to 0 to restart the process of tracking.

Once we found the points (X_o, X'_o) in the images, we calculated the lines of retroprojection.

$$L = X_{o_c} \wedge C_c \wedge e_\infty \tag{153}$$

$$L' = X'_{o_c} \wedge C_c \wedge e_\infty. \tag{154}$$

The point in 3D is the intersection of these lines. However, in the case when they do not intersect, the expected point will be considered as the center of their direct distance.

9.6. Smooth tracking

In this experiment, the binocular head should smoothly track a target. Figures 24(a), 25(a), and 26(a) show the 3D coordinates of the focus of attention. Figures 24(b), 25(b),

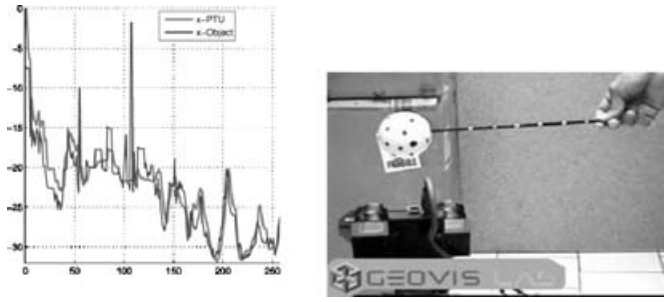

Fig. 24. (a) The x coordinate of the focus of attention. (b) The image of tracking.

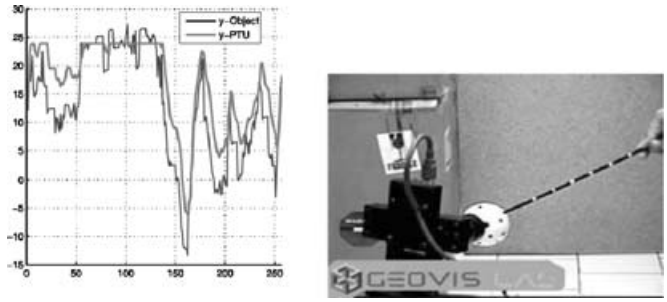

Fig. 25. (a) The y coordinate of the focus of attention. (b) The image of tracking.

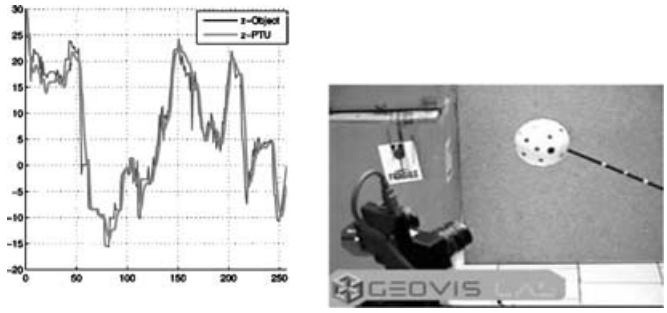

Fig. 26. (a) The z coordinate of the focus of attention. (b) The image of tracking.

and 26(b) show examples of the image sequence. We can see that the curves of the 3D object trajectory are very rough; however, the control rule manages to keep the trajectory of the pan–tilt unit smooth.

10. Conformal Geometric Control for Object Manipulation

This section shows a conformal geometric control technique used for robot manipulation of objects. This uses the inverse kinematics of the robot arm. The geometric control scheme is shown in Fig. 27. Note that the stereo system for image acquisition and the 3D reconstruction is represented by the L and R blocks.

Unlike a standard control technique that uses error of vector, we use as reference circles instead of points for computing the error in the 3D rigid transformation, which is coded efficiently using motors.

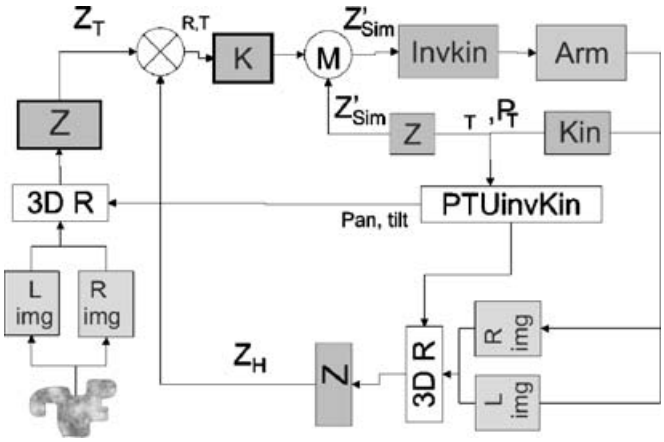

Fig. 27. Control scheme.

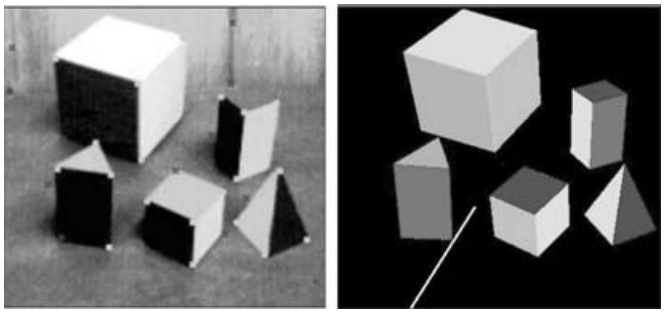

Fig. 28. (a) Corners in the image of the object. (b) 3D reconstruction of the object.

Next, we compute the grasping circle in the object. Then, the griper circle is estimated. Finally, the error in the translation and rotation (Motor M) is computed.

10.1. Grasping circle

The object in 3D is observed using the stereo-vision system of the robot; then key corners are automatically detected in each of the stereo images. Using triangulation, the robot computer computes a 3D reconstruction of the object [Fig. 28(b)].

Just four points of the object are considered for the computations. Three of them are in the base (x_1, x_2, x_3) and one more (x_4) is at the top of the object as we can see in Fig. 28(a). Using these points, one computes the grasping circle. The points of the base give us a circle z_b from which one computes the base plane π_b as follows:

$$z_b^* = x_1 \wedge x_2 \wedge x_3, \tag{155}$$

$$\pi_b = (z_b^* \wedge e_\infty) I_c. \tag{156}$$

The object might be grasped in the middle, which means the grasping circle is the translation of the circle z_b in the direction of x_4 as

$$T = 1 + \frac{1}{4}(\pi_b \cdot x_4)\pi_b e_\infty \tag{157}$$

$$z_t = T z_b \widetilde{T}. \tag{158}$$

Now, the grasping circle z_t will be the target.

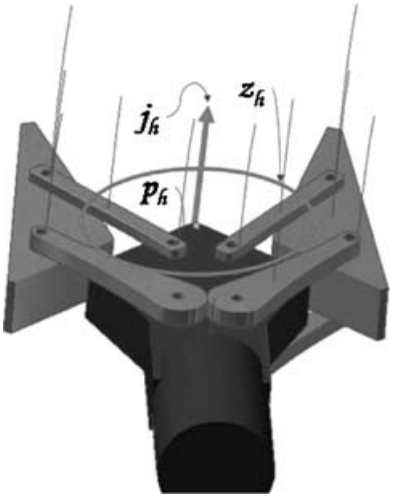

Fig. 29. Griper circle.

10.2. Griper circle

Similar to the grasping circle, the griper circle is estimated on the basis of the tracked screws of the griper that are seen in the pair of images. Figure 29 shows the position of the point p_h. This point is the center of the circle and is tracked with the cameras. The griper circle can be calculated easily, by creating a sphere with center at p_t and radius ρ equal to the middle of the aperture of the griper.

$$S_h = p_h - \frac{1}{2}\rho^2 e_\infty. \tag{159}$$

Now tracking two more points a and b on the griper, we create the plane π_h as

$$\pi_h = p_h \wedge a \wedge b \wedge e_\infty. \tag{160}$$

Finally, the griper circle is computed as

$$z_h = S_h \wedge \pi_h. \tag{161}$$

10.3. Motor estimation

In order to close the loop of control, we calculate the error between the griper circle and the grasping circle. This is achieved by simultaneously computing the 3D rotation and translation in terms of a motor (Fig. 30). The translation axis is computed easily having the centers of the circles. First, we calculate spheres with centers at each circle as

$$S_h = \frac{z_h^*}{z_h^* \wedge e_\infty} \tag{162}$$

$$S_t = \frac{z_t^*}{z_t^* \wedge e_\infty}. \tag{163}$$

Now, the translation axis is given by

$$l_T^* = s_h \wedge s_t \wedge e_\infty. \tag{164}$$

The distance d between circles is $d = |l^*|$. The rotation axis is computed using the axes of each circle l_h^* and l_t^*

$$l_h^* = z_h \wedge e_\infty \tag{165}$$

$$l_t^* = z_t \wedge e_\infty. \tag{166}$$

The axes l_h and l_t lying on the plane π_{th}^* are given by

$$\pi_{th}^* = l_t^* \wedge (l_h^* E). \tag{167}$$

Therefore, the rotation axis is

$$l_r^* = s_h \wedge \pi_{th} \wedge e_\infty. \tag{168}$$

The angle θ between circles is computed as follows:

$$\cos(\theta) = \frac{l_t^* \cdot l_h^*}{|l_t^*||l_h^*|}. \tag{169}$$

After that, we estimate the rotation and translation axes, and calculate the next position of the griper. This movement will reduce the error between the circles.

$$R = e^{-\frac{1}{2}\Delta\theta l_r} \tag{170}$$

$$T = 1 + \frac{1}{2}\Delta d l_t e_\infty \tag{171}$$

$$z_h' = T R z_h \widetilde{R}\widetilde{T}. \tag{172}$$

The circle z_h' is the new circle of the griper.

In Fig. 31(a), we can see two overlapped gripers. One of them is the griper in the real position [Fig. 31(b)] and the other one is the next position of the griper. The images of the griper are processed to extract the features and the vision system tracks them to know the position of the griper in real time.

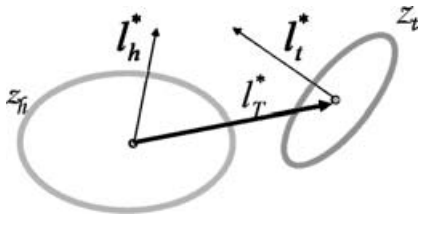

Fig. 30. Geometric relation between the griper circle z_h^* and the grasping circle z_t^*.

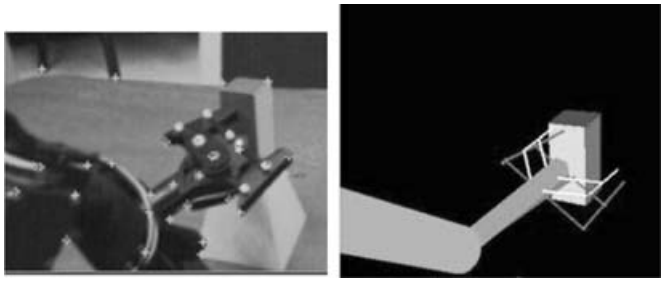

Fig. 31. (a) Tracking of the griper. (b) Real-time graphical result.

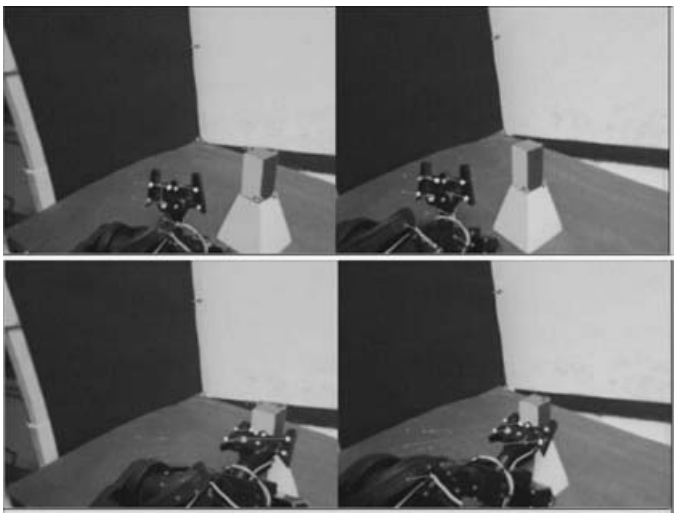

Fig. 32. Tracking of the griper with a stereoscopic vision system: The start position (*top*) and the final position (*left*).

Finally, in Fig. 32, the pair of images that are needed to retroproject the corners for computing the 3D reconstruction of the object and the griper, are presented. The images show the initial and the final positions of the griper. Note the lines of the motion.

11. Conclusion

In this paper, we have advantageously utilized the mathematical system of the conformal geometric algebra. We compute the inverse and differential kinematics of robot devices using a language of spheres showing how we simplify the complexity of the computations. In this work, we introduced a new geometric Jacobian in terms of bivectors, which is by far more effective in its representation as the standard Jacobian because its derivation is done in terms of the projections of the involved points onto the line axes. Furthermore, our Jacobian can be used for any kind of robot joints.

In this framework, we solve various tasks of 3D object manipulation, which is assisted by stereo-vision. All these computations are carried out using real images captured by a robot binocular head, and the manipulation is done by a five DOF robot arm mounted on a mobile robot. In this context, an interesting conformal geometric control technique is introduced. We present a very elegant application of our geometric Jacobian for smooth control of a binocular head.

We believe that the framework of conformal geometric algebra can be in general of great advantage for solving various complex tasks of visually guided robotics.

References

1. N. Andreff, Asservissement visuel à partir de droites et auto-étalonnage pince-camera *Ph.D. Thesis* (Grenoble, France: Institut National Polytechnique de Grenoble, 1999).
2. N. A. Aspragathos and J. K. Dimitros, "A comparative study of three methods for robot kinematics," *IEEE Trans. Syst., Man, Cybern.–B, Cybern.* **28**(2), 135–145, Apr. 1998.
3. E. Bayro-Corrochano, *Geometric Computing for Perception Action Systems* (Springer Verlag, Boston, MA, 2001).
4. E. Bayro-Corrochano, K. Daniilidis and G. Sommer, "Motor algebra for 3D kinematics. The case of the hand–eye calibration," *J. Math. Imaging Vis.* **13**(2), 79–100, 2000.
5. E. Bayro-Corrochano and D. Kähler, "Motor algebra approach for computing the kinematics of robot manipulators," *J. Robot. Syst.* **9**(17), 495–516, 2000.
6. E. Bayro-Corrochano and L. E. Falcón, "Geometric algebra of points, lines, planes and spheres for computer vision and robotics," *Robotica* **23**(6), 755–770, 2005.
7. C. Bregler and J. Malik, "Tracking people with twists and exponential maps," *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, Santa Barbara, CA (Jun. 23–25, 1998) pp. 8–15.
8. H. H. Chen, "A screw motion approach to uniqueness analysis of head-eye geometry," *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, Maui, HI (Jun. 3–6, 1991) pp. 145–151.
9. J. C. K. Chou and M. Kamel, "Finding the position and orientation of a sensor on a robot manipulator using quaternions," *Int. J. Robot. Res.* **10**(3), 240–254 (1991).
10. J. Denavit and R. S. Hartenberg, "A kinematic notation for the lower-pair mechanism based on matrices," *ASME J. Appl. Mech.* **77**, 215–221 (1955).
11. J. Funda and R. P. Paul, "A computational analysis of screw transformation in robotics," *IEEE Trans. Robot. Autom.* **6**(3), 348–356 (Jun. 1990).
12. Y. L. Gu and J. Y. S. Luh, "Dual-number transformation and its applications to robotics," *IEEE J. Robot. Autom.* **RA-3**(6), 615–623 (1987).
13. D. Hestenes, *Space–Time Algebra* (Gordon and Breach, New York, 1966).
14. D. Hestenes and G. Sobczyk, *Clifford Algebra to Geometric Calculus: A Unified Language for Mathematics and Physics* (Reidel, Amsterdam, The Netherlands, 1984).
15. D. Hestenes, H. Li and A. Rockwood, "New Algebraic Tools for Classical Geometry," In: *Geometric Computing with Clifford Algebra* (G. Sommer, ed.) (Springer-Verlag, Berlin Germany, 2001) pp. 3–23, Chap. I.
16. D. Hestenes and R. Ziegler, "Projective Geometry with Clifford Algebra," *Acta Appl. Math.* **23**, 25–63 (1991).
17. J. H. Kim and V. R. Kumar, "Kinematics of robot manipulators via line transformations," *J. Robot. Syst.* **7**(4), 649–674 (1990).
18. M. Li and D. Betsis, "Hand–eye calibration," *Proceedings of the International Conference on Computer Vision*, Boston, MA (Jun. 20–23, 1995) pp. 40–45.
19. J. M. McCarthy, Dual orthogonal matrices in manipulator kinematics," *Int. J. Robot. Res.* **5**(2), 45–51 (1986).
20. G. R. Pennock and A. T. Yang, "Application of dual-number matrices to the inverse kinematics problem of robot manipulators," *J. Mech. Transm. Autom. Des.* **107**, 201–208 (1985).
21. R. Sabata and J. K. Aggarwal, "Estimation of motion from a pair of range images: A review," *CVGIP, Image Underst.* **54**, 309–324 (1991).
22. Y. C. Shiu and S. Ahmad, "Calibration of wrist-mounted robotic sensors by solving homogeneous transform equations of the form $AX = XB$," *IEEE Trans. Robot. Autom.* **5**, 16–29 (1989).
23. R. Y. Tsai and R. K. Lenz, "A new technique for fully autonomous and efficient 3D robotics hand/eye calibration," *IEEE Trans. Robot. Autom.* **5**, 345–358 (1989).
24. M. W. Walker, "Manipulator kinematics and the epsilon algebra," *IEEE J. Robot. Autom.* **4**(2), 358–364 (1987).