Applied Bionics and Biomechanics 8 (2011) 411–428
DOI 10.3233/ABB-2011-0005
IOS Press

Robot object manipulation using stereoscopic vision and conformal geometric algebra

Julio Zamora-Esquivel[a,*] and Eduardo Bayro-Corrochano[b]
[a]*Intel, Jalisco, Mexico*
[b]*Department of Electrical Engineering and Computer Science, CINVESTAV, Unidad Guadalajara, Jalisco, Mexico*

Abstract. This paper uses geometric algebra to formulate, in a single framework, the kinematics of a three finger robotic hand, a binocular robotic head, and the interactions between 3D objects, all of which are seen in stereo images. The main objective is the formulation of a kinematic control law to close the loop between perception and actions, which allows to perform a smooth visually guided object manipulation.

Keywords: Conformal geometry, kinematics, grasping, tracking

1. Introduction

For the study of kinematics of mechanisms, different mathematical frameworks have been used such as vector calculus, quaternion algebra, or linear algebra, the last one being the most frequently used. However, in these mathematical systems it is very complicated to handle the kinematics and dynamics involving geometric primitives like points, lines, and planes. This paper shows how the mathematical treatment is much easier when it is handled in the conformal geometric algebra framework. In order to exemplify, the formulation of the kinematics for a binocular robot head and for the Barrett hand are presented. In order to close the loop between perception and action, the pose of the object and the hand are estimated first, and then a control law for approaching and grasping is applied. This control law is geometrically formulated using the visual-mechanical Jacobian matrix, which in turn is computed using the principal lines of the camera and the axis of the pan-tilt unit. In addition, it is shown how to obtain a feasible grasping strategy based in the mathematical model of the object and the manipulator.

2. Geometric algebra: An outline

Let G_n denote the geometric algebra of n-dimensions, which is a graded-linear space. As well as vector-addition and scalar multiplication, there is a non-commutative product, which is associative and distributive over addition. This is the *geometric* or *Clifford product*.

The inner product of two vectors is the standard *scalar* or *dot* product, which produces a scalar. The outer or wedge product of two vectors is a new quantity which we call a *bivector*. A bivector is thought of as an oriented area in the plane containing a and b, which is formed by sweeping a along b.

Thus, $b \wedge a$ will have the opposite orientation, making the wedge product anti-commutative. The outer product is immediately generalizable to higher dimensions. For example, $(a \wedge b) \wedge c$, a *trivector*, is interpreted as the oriented volume formed by sweeping the area $a \wedge b$ along vector c. The outer product of k vectors is a k-blade, and such a quantity is said to

*Corresponding author. Email: julio.zamora.esquivel@intel.com.

have *grade k*. A *multivector* the (linear combination of objects of different grades) is a *homogeneous k*-vector if it contains terms of only a single grade *k*.

In this paper, the geometric algebra G_n of the n dimensional space by $G_{p,q,r}$ will be specified, where p, q and r stand for the number of basis vectors which square to 1, -1 and 0 respectively and fulfill $n = p + q + r$.

e_i will be used to denote basis vector i. In the geometric algebra $G_{p,q,r}$, the geometric product of two basis vectors is defined as

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

This leads to a basis for the entire algebra:

$$\{1\}, \{e_i\}, \{e_i \wedge e_j\}, \{e_i \wedge e_j \wedge e_k\}, \ldots, \{e_1 \wedge e_2 \wedge \ldots \wedge e_n\} \quad (1)$$

Any multivector can be expressed in terms of this basis. The multivectors can be of grade 0 (scalars), grade 1 (vectors), grade 2 (bivectors), grade 3 (trivectors), etc., up to grade n (n-vectors). For example, $G_{4,1,0}$ has the basis

$$\{1\}, \{e_1, \ldots, e_5\}, \{e_{12}, e_{13}, \ldots, e_{45}\}, \{e_{123}, \ldots, e_{345}\}, \{e_{1234}, \ldots, e_{2345}\}, \{e_{12345} = I\} \quad (2)$$

where $e_1^2 = 1, e_2^2 = 1, e_3^2 = 1, e_4^2 = 1, e_5^2 = -1$. $G_{4,1,0}$ is a five-dimensional geometric algebra with $2^5 = 32$ multivector blades.

3. Conformal geometry

Geometric algebra $G_{4,1} = G_{4,1,0}$ can be used to treat conformal geometry in a very elegant way. To see how this is possible, the same formulation presented in [1] is followed, and the Euclidean vector space $\mathbb{R}^3$ is represented in $\mathbb{R}^{4,1}$. This space has an orthonormal vector basis given by $\{e_i\}$. $e_{ij} = e_i \wedge e_j$ are bivectorial bases. Bivector basis e_{23}, e_{31}, and e_{12} correspond together with 1 to Hamilton's quaternions. The Euclidean pseudo-scalar unit $I_e := e_1 \wedge e_2 \wedge e_3$, a pseudo-scalar $I = I_e E$, and the bivector $E := e_4 \wedge e_5 = e_4 e_5$ are used for computing Euclidean and conformal duals of multivectors. For more about conformal geometric algebra, refer to [1, 3, 4].

3.1. The stereographic projection

Conformal geometry is related to a stereographic projection in Euclidean space. A stereographic projection consists on a mapping the points lying on a hypersphere to points lying on a hyperplane. In this case, the projection plane passes through the equator, and the sphere is centered at the origin. To make a projection, a line is drawn from the north pole to each point on the sphere and the intersection of this line, where with the projection plane constitutes the stereographic projection.

For simplicity, we will illustrate the equivalence between stereographic projections and conformal geometric algebra of $\mathbb{R}^1$. We will be working in $\mathbb{R}^{2,1}$ with the basis vectors $\{e_1, e_4, e_5\}$ having the above mentioned properties. The projection plane will be the x-axis, and the sphere will be a circle centered at the origin with unitary radius.

Given a scalar x_e representing a point on the x-axis, point x_c, lying on the circle that projects to it, is calculated (see Fig. 1). The equation of the line passing through the north pole and x_e is given as $f(x) = -\frac{1}{x_e}x + 1$, and the equation of the circle is $x^2 + g(x)^2 = 1$. Substituting the equation of the line on the circle $g = f$, the point of intersection x_c is obtained, which can be represented in homogeneous coordinates as the vector

$$x_c = 2\frac{x_e}{x_e^2 + 1}e_1 + \frac{x_e^2 - 1}{x_e^2 + 1}e_4 + e_5. \quad (3)$$

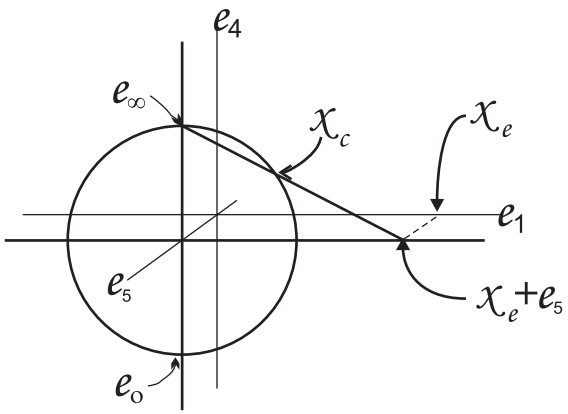

Fig. 1. Stereographic projection for 1-D.

From (3) we can infer the coordinates on the circle for the point at infinity as

$$\begin{aligned} e_\infty &= \lim_{x_e\to\infty}\{x_c\} \\ &= \lim_{x_e\to\infty}\left\{2\tfrac{x_e}{x_e^2+1}e_1 + \tfrac{x_e^2-1}{x_e^2+1}e_4 + e_5\right\} \\ &= e_4 + e_5, \end{aligned} \tag{4}$$

$$\begin{aligned} e_o &= \tfrac{1}{2}\lim_{x_e\to 0}\{x_c\} \\ &= \tfrac{1}{2}\lim_{x_e\to 0}\left\{2\tfrac{x_e}{x_e^2+1}e_1 + \tfrac{x_e^2-1}{x_e^2+1}e_4 + e_5\right\} \\ &= \tfrac{1}{2}(-e_4 + e_5), \end{aligned} \tag{5}$$

Note that (3) can be rewritten as

$$x_c = x_e + \frac{1}{2}x_e^2 e_\infty + e_o, \tag{6}$$

3.2. Spheres and planes

The equation of a sphere of radius ρ centered at point $p_e \in \mathbb{R}^3$ can be written as $(x_e - p_e)^2 = \rho^2$. Since $x_c \cdot y_c = -1/2(\mathrm{x}_e - \mathrm{y}_e)^2$, where x_e and y_e are the Euclidean components, and $x_c \cdot p_c = -1/2\rho^2$, the formula above can be rewritten in terms of homogeneous coordinates. Since $x_c \cdot e_\infty = -1$, the expression above can be factored to

$$x_c \cdot (p_c - \frac{1}{2}\rho^2 e_\infty) = 0, \tag{7}$$

This equation corresponds to the so called Inner Product Null Space (IPNS) representation, which finally yields the simplified equation for the sphere as $s = p_c - 1/2\rho^2 e_\infty$. Note from this equation that a point is just a sphere with a radius of zero. Alternatively, the dual of the sphere is represented as 4-vector $s^* = sI$. The advantage of the dual form is that the sphere can be directly computed from four points as

$$s^* = x_{c_1} \wedge x_{c_2} \wedge x_{c_3} \wedge x_{c_4}. \tag{8}$$

If one of these points are replaced for the point at infinity, the equation of a 3D plane is obtained

$$\pi^* = x_{c_1} \wedge x_{c_2} \wedge x_{c_3} \wedge e_\infty. \tag{9}$$

So that π becomes in standard IPNS form

$$\pi = I\pi^* = n + de_\infty \tag{10}$$

where n is the normal vector and d represents the Hesse distance for the 3D space.

3.3. Circles and lines

A circle z can be regarded as the intersection of two spheres s_1 and s_2 as $z = (s_1 \wedge s_2)$ in IPNS. The dual form of the circle can be expressed by three points lying on the circle, namely

$$z^* = x_{c_1} \wedge x_{c_2} \wedge x_{c_3}. \tag{11}$$

Similar to the case of planes, lines can be defined by circles passing through the point at infinity as:

$$L^* = x_{c_1} \wedge x_{c_2} \wedge e_\infty. \tag{12}$$

The standard IPNS form of the line can be expressed as

$$L = nI_e - e_\infty m I_e, \tag{13}$$

where n and m stand for the line orientation and moment, respectively. The line in the IPNS standard form is a bivector representing the six Plücker coordinates.

4. Rigid transformations

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

4.1. Reflection

The combination of reflections of conformal geometric entities enables the forming of other transformations. The reflection of a point x with respect to the plane π is equal to x minus twice the directed distance between the point and plane (see Fig. 2). That is, $ref(x) = x - 2(\pi \cdot x)\pi^{-1}$. This expression is calculated by using the reflection $ref(x_c) = -\pi x_c \pi^{-1}$ and the property of Clifford product of vectors $2(b \cdot a) = ab + ba$.

Table 1
Representation of conformal geometric entities in conformal geometric algebra

Entity	IPNS Representation	OPNS Dual representation
Sphere	$s = p - 1/2\rho^2 e_\infty$	$s^* = x_1 \wedge x_2 \wedge x_3 \wedge x_4$
Point	$x_c = x_e + 1/2x_e^2 e_\infty + e_0$	$x^* = s_1 \wedge s_2 \wedge s_3 \wedge s_4$
Line	$L = nI_e - e_\infty m I_e$	$L^* = x_1 \wedge x_2 \wedge e_\infty$
Plane	$\pi = n + de_\infty$	$\pi^* = x_1 \wedge x_2 \wedge x_3 \wedge e_\infty$
Circle	$z = s_1 \wedge s_2$	$z^* = x_1 \wedge x_2 \wedge x_3$
Pair of P	$P_p = s_1 \wedge s_2 \wedge s_3$	$P_p^* = x_1 \wedge x_2$

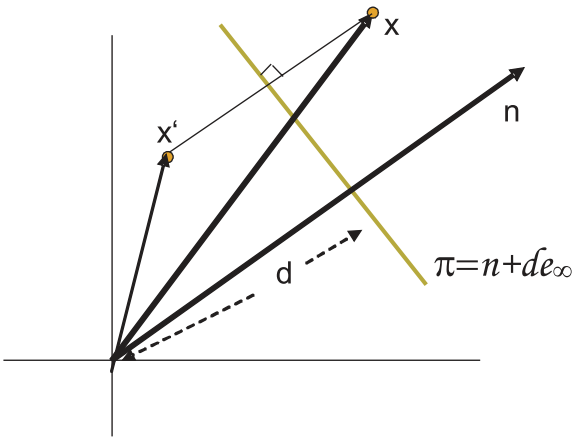

Fig. 2. Reflection of a point x with respect to the plane π.

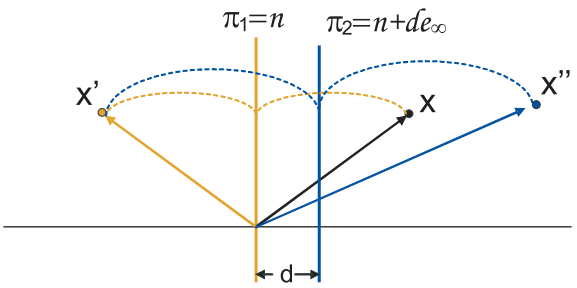

Fig. 3. Reflection about parallel planes.

For a IPNS geometric entity Q, the reflection with respect to the plane π is given as

$$Q' = \pi Q \pi^{-1} \tag{14}$$

4.2. Translation

The translation of conformal geometric entities can be done by carrying out two reflections at parallel planes π_1 and π_2 (see Fig. 3). That is

$$Q' = \underbrace{(\pi_2\pi_1)}_{T_a} Q \underbrace{(\pi_1^{-1}\pi_2^{-1})}_{\widetilde{T}_a} \tag{15}$$

$$T_a = (n + de_\infty)n = 1 + \frac{1}{2}ae_\infty = e^{\frac{a}{2}e_\infty} \tag{16}$$

with $a = 2dn$.

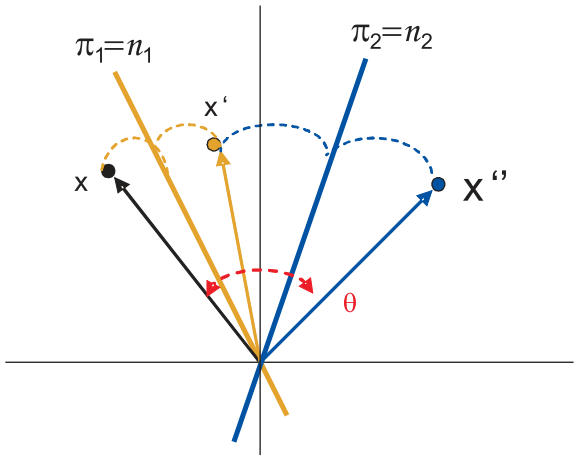

Fig. 4. Reflection about nonparallel planes.

4.3. Rotation

The rotation is the product of two reflections at nonparallel planes which pass through the origin (see Fig. 4)

$$Q' = \underbrace{(\pi_2\pi_1)}_{R_\theta} Q \underbrace{(\pi_1^{-1}\pi_2^{-1})}_{\widetilde{R}_\theta} \tag{17}$$

Or computing the conformal product of the normals of the planes.

$$R_\theta = n_2 n_1 = cos\left(\frac{\theta}{2}\right) - sin\left(\frac{\theta}{2}\right) l = e^{-\frac{\theta}{2}l} \tag{18}$$

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

$$Q' = \underbrace{(TR\widetilde{T})}_{M_\theta} Q \underbrace{(T\widetilde{R}\widetilde{T})}_{\widetilde{M_\theta}} \tag{19}$$

$$M_\theta = TR\widetilde{T} = cos\left(\frac{\theta}{2}\right) - sin\left(\frac{\theta}{2}\right) L = e^{-\frac{\theta}{2}L} \tag{20}$$

The direct kinematics for serial robot arms is a succession of motors as can be seen in [2], and it is valid for points, lines, planes, circles, and spheres

$$Q' = \prod_{i=1}^{n} M_i Q \prod_{i=1}^{n} \widetilde{M}_{n-i+1} \tag{21}$$

5. Barrett hand direct kinematics

In this work, the Barret hand is used, which is a type of spherical hand. This hand has three fingers, which are moved around a circle mimicking a five to six fingered hand. It was found quite convenient to formulate the kinematics of the Barret in the conformal geometric algebra framework because this mathematical system offers a very powerful geometric language, which uses as computational geometric unit the sphere. The grasping of this spherical hand can be formulated in an intuitive and easy way in terms of geometric entities like lines, planes, circles, and spheres, see [3].

The direct kinematics involves the computation of the position and orientation of the end-effector given the parameters of the joints. The direct kinematics can be easily computed given the lines of the screws.

In order to explain the kinematics of Barrett hand, the kinematic of one finger is shown. This example will assume that the finger is totally extended. Note that such a hypothetical positions is not reachable in normal operation, but it simplifies the explanation.

We initiated denoting some points on the finger, which help describe their position.

$$\mathrm{x}_{1o} = A_w e_1 + A_1 e_2 + D_w e_3, \tag{22}$$

$$\mathrm{x}_{2o} = A_w e_1 + (A_1 + A2) e_2 + D_w e_3, \tag{23}$$

$$\mathrm{x}_{3o} = A_w e_1 + (A_1 + A_2 + A_3) e_2 + D_w e_3. \tag{24}$$

The points x_{1o}, x_{2o}, and x_{3o} describe the position of each joint and the end of the finger in the Euclidean space, see Fig. 5.

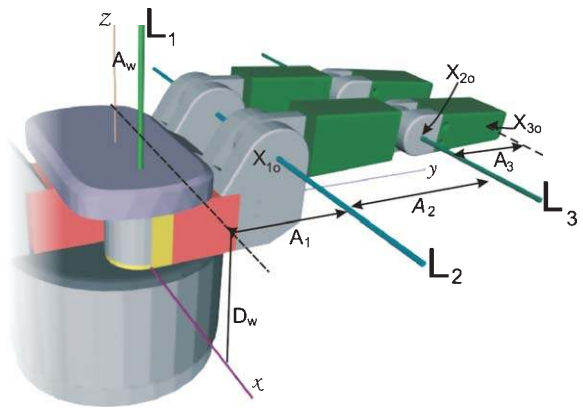

Fig. 5. Barrett hand hypothetical position.

Each one of these points is mapped to a point of conformal geometry by $x = \mathrm{x} + 1/2\,\|\mathrm{x}\|^2 e_\infty + e_o$

Once these points have been defined, it is quite simple to calculate the axis lines

$$L_{1o} = -A_w(e_2 \wedge e_\infty) + e_{12}, \tag{25}$$

$$L_{2o} = (x_{1o} \wedge e_1 \wedge e_\infty)\, I_c, \tag{26}$$

$$L_{3o} = (x_{2o} \wedge e_1 \wedge e_\infty)\, I_c. \tag{27}$$

which will then be used as motor's axes. When the hand is initialized, the fingers move away to home position. That means $\Phi_2 = 2.46^o$ in the joint two and $\Phi_3 = 50^o$ degrees in joint three. In order to move the finger from this hypothetical position to its home position, an appropriate transformation needs to be obtained

$$M_{2o} = cos\,(\Phi_2/2) - sin(\Phi_2/2) L_{2o}, \tag{28}$$

$$M_{3o} = cos\,(\Phi_3/2) - sin(\Phi_3/2) L_{3o}. \tag{29}$$

Once the transformations have been calculated, they are applied to the points and lines, which must move.

$$x_2 = M_{2o} x_{2o} \widetilde{M}_{2o}, \tag{30}$$

$$x_3 = M_{2o} M_{3o} x_{3o} \widetilde{M}_{3o} \widetilde{M}_{2o}, \tag{31}$$

$$L_3 = M_{2o} L_{3o} \widetilde{M}_{2o}. \tag{32}$$

The point $x_1 = x_{1o}$ is not affected by the transformation, the same for the lines $L_1 = L_{1o}$ and $L_2 = L_{2o}$ (see Fig. 6).

Since the rotation angle of both axis L_2 and L_3 are related, fractions of angle q_1 will be used to describe their individual rotation angles. The motors of each joint are computed using $(2/35)2q_4$ to rotate around L_1, $(1/125)q_1$ around L_2 and $(1/375)q_1$ around L_3, the angles' coefficients where taken from the Barrett hand user manual.

$$M_1 = cos(q_4/35) + sin(q_4/35) L_1, \tag{33}$$

$$M_2 = cos(q_1/250) - sin(q_1/250) L_2, \tag{34}$$

$$M_3 = cos(q_1/750) - sin(q_1/750) L_3. \tag{35}$$

The position of each point is related to the angles q_1 and q_4 as follows:

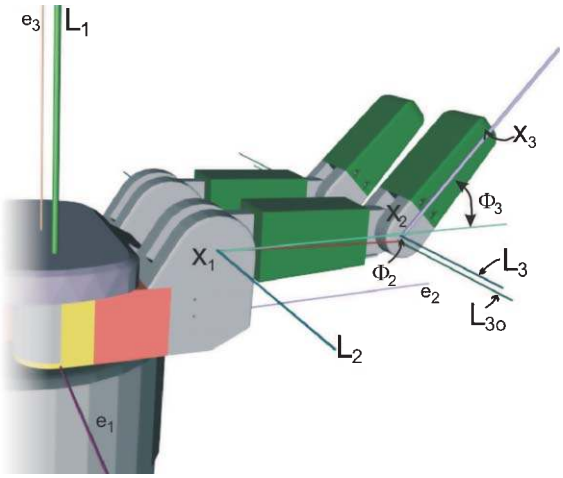

Fig. 6. Barrett hand at home position.

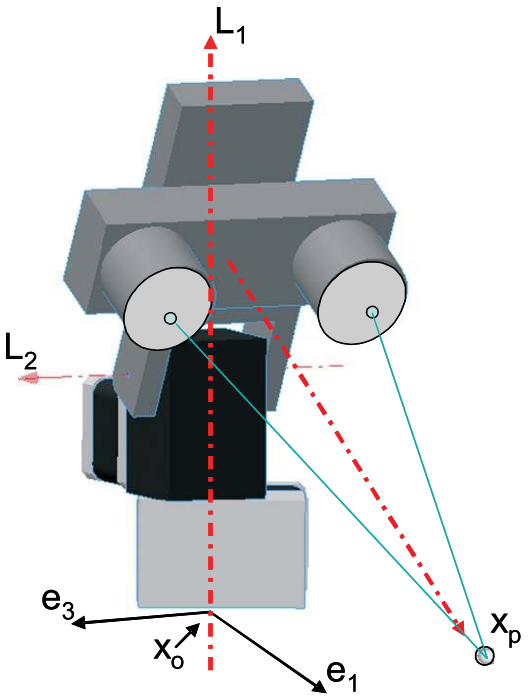

Fig. 7. Pan tilt unit has 2 D.O.F. and the depth as a virtual D.O.F.

$$x'_1 = M_1 x_1 \widetilde{M}_1, \tag{36}$$

$$x'_2 = M_1 M_2 x_2 \widetilde{M}_2 \widetilde{M}_1, \tag{37}$$

$$x'_3 = M_1 M_2 M_3 x_3 \widetilde{M}_3 \widetilde{M}_2 \widetilde{M}_1. \tag{38}$$

$$L'_3 = M_1 M_2 L_3 \widetilde{M}_2 \widetilde{M}_1, \tag{39}$$

$$L'_2 = M_1 L_2 \widetilde{M}_1. \tag{40}$$

Since x'_3, L'_1, L'_2 and L'_3 are already known, the speed of the end of the finger can be calculated as follows

$$\dot{X}'_3 = X'_3 \cdot \left(-\frac{2}{35} L'_1 \dot{q}_4 + \frac{1}{125} L'_2 \dot{q}_1 + \frac{1}{375} L'_3 \dot{q}_1 \right). \tag{41}$$

where $\dot{q}_1$ and $\dot{q}_4$ represent the velocity of its respective joint.

6. Tracking

An example using our new formulation of the Jacobian will be presented. This is the control of a pan-tilt unit.

6.1. The pan-tilt unit

An algorithm for the velocity control of a pan-tilt unit (PTU Fig. 7) is implemented assuming three degree of freedom. The stereo depth is considered as one virtual D.O.F., thus the PTU has a similar kinematic behavior as a serial robot arm with three D.O.F.

In order to carry out a velocity control, the direct kinematics need to be computed first. This is very easy to do, the axis lines are known:

$$L_1 = -e_{31} \tag{42}$$

$$L_2 = e_{12} + d_1 e_1 e_\infty \tag{43}$$

$$L_3 = e_1 e_\infty \tag{44}$$

Since $M_i = e^{-\frac{1}{2} q_i L_i}$ and $\widetilde{M}_i = e^{\frac{1}{2} q_i L_i}$, the position of end effectors is computed as

$$x_p(q) = x'_p = M_1 M_2 M_3 x_p \widetilde{M}_3 \widetilde{M}_2 \widetilde{M}_1. \tag{45}$$

The state variable representation of the system is as follows

$$\begin{cases} \dot{x}'_p = x' \cdot (L'_1 \; L'_2 \; L'_3) \begin{pmatrix} u_1 \\ u_2 \\ u_3 \end{pmatrix} \\ y = x'_p \end{cases} \tag{46}$$

where the position of 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. 6), the line L'_i is the current position of L_i and u_i is the velocity of the i-joint of the system. As L_3 is an axis at infinity M_3 is a translator, that is, the virtual component is a prismatic joint.

6.2. Exact linearization via feedback

The following state feedback control law is now chosen in order to get a new linear system.

$$\begin{pmatrix} u_1 \\ u_2 \\ u_3 \end{pmatrix} = (x'_p \cdot L'_1 \; x'_p \cdot L'_2 \; x'_p \cdot L'_3)^{-1} \begin{pmatrix} v_1 \\ v_2 \\ v_3 \end{pmatrix} \tag{47}$$

where $V = (v_1, v_2, v_3)^T$ is the new input to the linear system. Then, the equations of the system are rewritten

$$\begin{cases} \dot{x}'_p = V \\ y = x'_p. \end{cases} \tag{48}$$

6.3. Asymptotic output tracking

The problem of following a constant reference x_t is solved computing the error between the end effector's 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

$$V = -ke \tag{49}$$

This error is small if the control system is doing it's job. The actual error is mapped to an error in the joint space using the Jacobian inverse.

$$U = J^{-1}V \tag{50}$$

where the Jacobian is expressed as $J = x'_p \cdot (L'_1 \; L'_2 \; L'_3)$

$$\begin{aligned} j_1 &= x'_p \cdot (L_1) \\ j_2 &= x'_p \cdot (M_1 L_2 \widetilde{M}_1) \\ j_3 &= x'_p \cdot (M_1 M_2 L_3 \widetilde{M}_2 \widetilde{M}_1) \end{aligned} \tag{51}$$

Once the Jacobian is calculated, it is easy to compute the dq_i using crammer's rule in terms of wedge products.

$$\begin{pmatrix} u_1 \\ u_2 \\ u_3 \end{pmatrix} = (j_1 \wedge j_2 \wedge j_3)^{-1} \cdot \begin{pmatrix} V \wedge j_2 \wedge j_3 \\ j_1 \wedge V \wedge j_3 \\ j_1 \wedge j_2 \wedge V \end{pmatrix} \tag{52}$$

This is possible because $j_1 \wedge j_2 \wedge j_3 = det(J) I_e$. Finally we have dq_i which will tend to reduce these errors. This control law could be used in case of $j_1 \neq j_2 \neq j_3$, otherwise the Jacobian matrix will have singularities. Then the pseudo inverse of Jacobian most be used.

6.4. Pseudo-inverse of the Jacobian

Since j_1 and j_2 are tree dimensional vectors, they generate a 3×2 Jacobian matrix. In this case the pseudo inverse of Jacobian matrix is being used to create a new control law

$$J = \begin{bmatrix} j_1 \; j_2 \end{bmatrix} \tag{53}$$

Using the pseudo-inverse of Moore-Penrose

$$J^+ = (J^T J)^{-1} J^T \tag{54}$$

Now evaluating J in 54

$$J^+ = \frac{1}{det(J^T J)} \begin{pmatrix} (j_2 \cdot j_2) j_1 - (j_2 \cdot j_1) j_2 \\ (j_1 \cdot j_1) j_2 - (j_2 \cdot j_1) j_1 \end{pmatrix} \tag{55}$$

Using Clifford algebra, this equation could be further simplified

$$\begin{aligned} det(J^T J) &= (j_1 \cdot j_1)(j_2 \cdot j_2) - (j_1 \cdot j_2)^2 && (56) \\ &= (|j_1||j_2|)^2 - (|j_1||j_2|)^2 cos^2(\theta), && (57) \\ &= (|j_1||j_2|)^2 sin^2(\theta), && (58) \\ &= |j_1 \wedge j_2|^2 && (59) \end{aligned}$$

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

$$(j_2 \cdot j_2) j_1 - (j_2 \cdot j_1) j_2 = j_2 \cdot (j_2 \wedge j_1) \tag{60}$$

$$(j_1 \cdot j_1) j_2 - (j_2 \cdot j_1) j_1 = j_1 \cdot (j_1 \wedge j_2) \tag{61}$$

Now the equation (54) can be rewritten as

$$\begin{aligned} J^+ &= \frac{1}{|j_1 \wedge j_2|^2} \begin{pmatrix} j_2 \cdot (j_2 \wedge j_1) \\ j_1 \cdot (j_1 \wedge j_2) \end{pmatrix} \\ &= \begin{pmatrix} j_2 \cdot (j_2 \wedge j_1)^{-1} \\ j_1 \cdot (j_1 \wedge j_2)^{-1} \end{pmatrix} \end{aligned} \tag{62}$$

Using this equation, the input can be computed as $U = J^{+}V$. That is equal to

$$U = (j_1 \wedge j_2)^{-1} \cdot \begin{pmatrix} V \wedge j_2 \\ j_1 \wedge V \end{pmatrix} \tag{63}$$

Note that in subsections 6.3 and 6.4, the Jacobian, its inverse, and pseudo-inverse was formulated using wedge products. This gives a geometric interpretation of the matrix columns. The existence of singularities is inherent to the problem, and it is independent of the mathematical formalism. However, the use of conformal geometric algebra helps formulate the entries of the Jacobian in terms of inner products of centers of mass and the involved line axes (see equation 51). The independence of the columns indicate the presence of singularities, which, due to our geometric representation, orient us directly towards the robot configuration we should expect a singularity from. The equation 52 is also an interesting and very useful equation where the entries of the Jacobian, with fully geometric interpretation, are combined with the control law V to compute the joint velocities. This way, the equation that relates the joint velocities with the velocities of projected 3D points will be derived below . This can be used to monitor the use of monocular vision motion in the 3D visual space.

6.5. *Visual tracking*

The target point is measured using two calibrated cameras (see Fig. 8). With each image, the center of

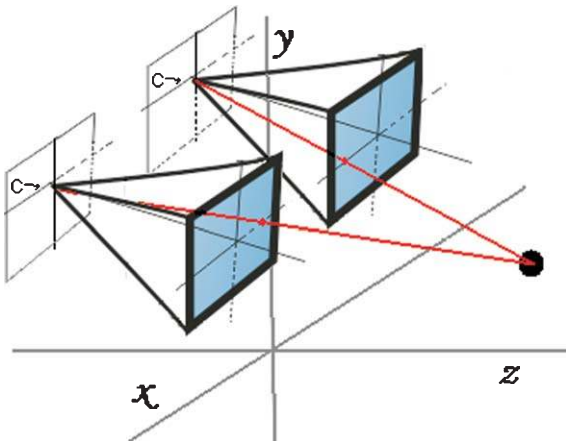

Fig. 8. Stereo cameras.

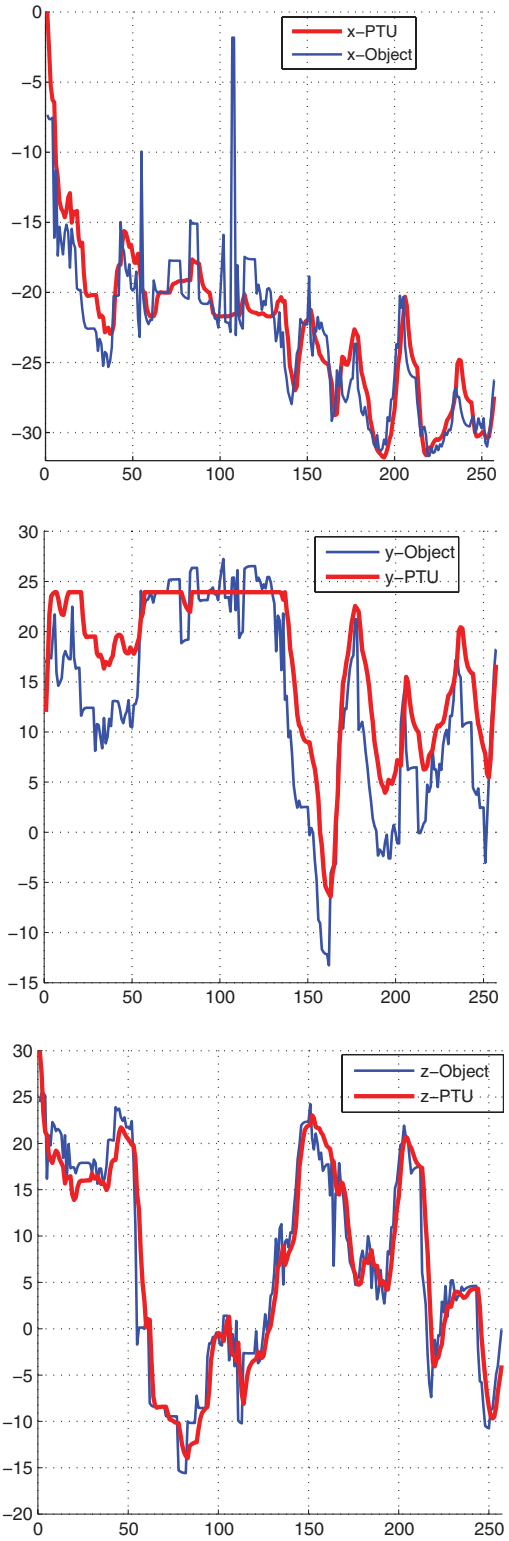

Fig. 9. (x, y, z) coordinates of the focus of attention.

mass of the object in movement is estimated and retro-projected in order to finally estimate its 3D point. To compute the mass center, the current image I_a is first subtracted to an image in memory I_b. The image in memory is the average of the last N images. This allows the background to be subtracted.

$$I_k(t) = I_a(t) - I_b(t-1) * N \tag{64}$$

$$I_b(t) = (I_b(t-1) * N + I_a)/(N+1) \tag{65}$$

Afterwards, 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. This is the way the mass center is calculated.

$$x_o = \frac{\int_0^n \int_0^m I_k y dx dy}{\int_0^n \int_0^m I_k dx dy} \tag{66}$$

$$y_o = \frac{\int_0^n \int_0^m I_k x dx dy}{\int_0^n \int_0^m I_k dx dy} \tag{67}$$

When the camera moves, the background changes. Then, it is necessary to reset N to 0 in order to restart the tracking process.

Once points (X_o, X'_o) are located in the images, the lines of retro-projection are calculated.

$$L = X_{o_c} \wedge C_c \wedge e_\infty, \tag{68}$$

$$L' = X'_{o_c} \wedge C_c \wedge e_\infty. \tag{69}$$

The 3D point is the intersection of these lines.

In tracking experiment, the binocular head should smoothly track a target. The Fig. (9) shows the $3D$ coordinates of the focus of attention. Figure 10 shows examples of the image sequence. 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.

6.6. Visual Jacobian

A point in the image is given by $s = (x, y)^T$. Whereas a 3-D point is represented by X. The relationship between $\dot{s}$ and $\dot{S}$ is called visual Jacobian.

The projection matrix of a camera in general position is represented by the planes π_1, π_2 y π_3.

$$P = \begin{pmatrix} \pi_1 \\ \pi_2 \\ \pi_3 \end{pmatrix}, \tag{70}$$

Point X is projected onto the image at the point

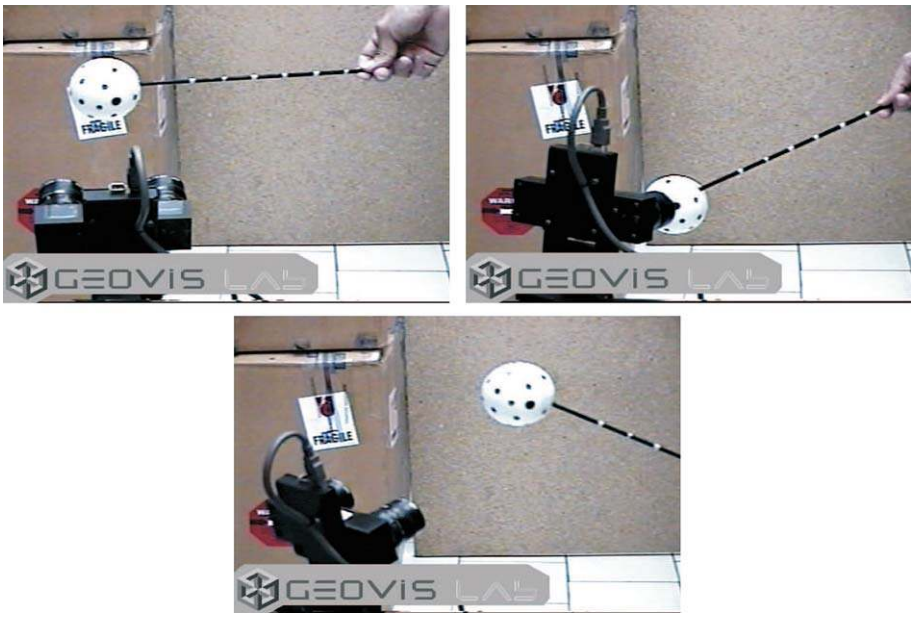

Fig. 10. Tracking sequence.

$$s = \begin{pmatrix} \frac{\pi_1 \cdot X}{\pi_3 \cdot X} \\ \frac{\pi_2 \cdot X}{\pi_3 \cdot X} \end{pmatrix} \tag{71}$$

To simplify the explanation, variable x is introduced and its time derivative $\dot{x}$ defined as

$$x = \begin{pmatrix} \pi_1 \cdot X \\ \pi_2 \cdot X \\ \pi_3 \cdot X \end{pmatrix} \quad \dot{x} = \begin{pmatrix} \pi_1 \cdot \dot{X} \\ \pi_2 \cdot \dot{X} \\ \pi_3 \cdot \dot{X} \end{pmatrix} \tag{72}$$

Furthermore s is given by $s_1 = x_1/x_3$ and its derivative

$$\dot{s}_1 = \dot{x}_1 \frac{1}{x_3} + x_1 \left(\frac{-\dot{x}_3}{x_3^2} \right) \tag{73}$$

$$\dot{s}_1 = \frac{x_3 \dot{x}_1 - x_1 \dot{x}_3}{x_3^2} \tag{74}$$

By substitution of x and $\dot{x}$ in equation 74 one gets

$$\dot{s}_1 = \kappa \left[(\pi_3 \cdot X) \pi_1 - (\pi_1 \cdot X) \pi_3 \right] \cdot \dot{X} \tag{75}$$

$$\dot{s}_1 = \kappa \left[X \cdot (\pi_3 \wedge \pi_1) \right] \cdot \dot{X} \tag{76}$$

where $\kappa = 1/x_3^2$. Following the same algebraic steps for s_2, it is possible to write the following equation

$$\dot{s} = \kappa X \cdot \begin{pmatrix} \pi_1 \wedge \pi_3 \\ \pi_2 \wedge \pi_3 \end{pmatrix} \cdot \dot{X} \tag{77}$$

Geometrically, $\pi_1 \wedge \pi_3$ represents an intersection line of planes π_1 and π_3. We denote the lines of these intersections by L_x and L_y

$$L_x = \pi_1 \wedge \pi_3 \tag{78}$$

$$L_y = \pi_2 \wedge \pi_3 \tag{79}$$

It is possible then to rewrite 77 as

$$\dot{s} = \kappa X \cdot \begin{pmatrix} L_x \\ L_y \end{pmatrix} \cdot \dot{X} \tag{80}$$

In order to close the loop between the perception and action, the relationship between velocities at the points of the image and the velocities of the joints of the pan-tilt unit are computed.

Taking the equation of differential kinematics 46 and visual Jacobian 80, it is possible to finally write a new expression

$$\dot{s} = \kappa \begin{pmatrix} \left(X' \cdot L'_x\right) \cdot \left(X' \cdot L'_1\right) & \left(X' \cdot L'_x\right) \cdot \left(X' \cdot L'_2\right) \\ \left(X' \cdot L'_y\right) \cdot \left(X' \cdot L'_1\right) & \left(X' \cdot L'_y\right) \cdot \left(X' \cdot L'_2\right) \end{pmatrix} \dot{q} \tag{81}$$

Equation 81 can be used to relate the visual world with the robot mechanism (pan-tilt unit). Since the velocities at the joints $\dot{q}_i$ can be estimated, the inverse differential kinematics of the mechanism can be computed backwards until the first joint is reached close to the camera. Then, by computing the inner product of the center of mass of the limbs with the joint axes L_i together with $\dot{q}_1$ of the first joint near to the camera, equation 81 can be computed. This equation for monocular vision follows the velocity of 3D points. Since the complexity of the expression is greatly reduced, and it is robust due to the use of inner products with line axes, the approach is fast and accurate. The associated control is linear, thus the error is minimized through time guaranteeing its convergence.

6.7. *Exact linearization via feedback*

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

$$\begin{pmatrix} u_1 \\ u_2 \end{pmatrix} = \begin{pmatrix} \left(X' \cdot L'_x\right) \cdot \left(X' \cdot L'_1\right) & \left(X' \cdot L'_x\right) \cdot \left(X' \cdot L'_2\right) \\ \left(X' \cdot L'_y\right) \cdot \left(X' \cdot L'_1\right) & \left(X' \cdot L'_y\right) \cdot \left(X' \cdot L'_2\right) \end{pmatrix}^{-1} \times \begin{pmatrix} v_1 \\ v_2 \end{pmatrix}$$

where $V = (v_1, v_2)^T$ is the new input to the linear system. The equations of the system are then rewritten

$$\begin{cases} \dot{s}'_p = V \\ y = s'_p \end{cases} \tag{82}$$

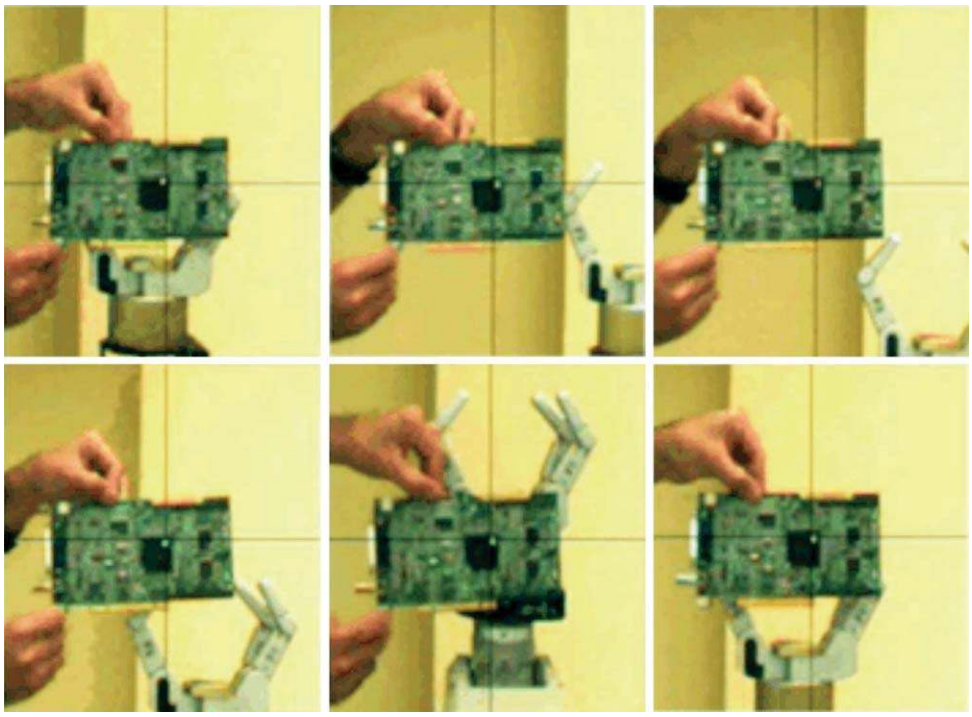

Fig. 11. Sequence of tracking.

For this test, the task of tracking of a circuit board was selected. The coordinate system was selected at the center of the camera, and the principal planes of the camera are given by

$$\pi_1 = f_x e_1 + x_o e_3 \tag{83}$$

$$\pi_2 = f_y e_2 + y_o e_3 \tag{84}$$

$$\pi_3 = e_3 \tag{85}$$

where f_x, f_y, x_o y y_o are the camera's parameters. Using these planes, lines L_x y L_y are computed. The axis of the pan-tilt is known.

$$L_1 = e_{23} + d_1 e_2 \tag{86}$$

$$L_2 = e_{12} + d_2 e_2 \tag{87}$$

Note that the tilt axis is called L_1, and the pan axis is L_2 because the coordinate system is in the camera. L_2' is a function of the tilt angle $L_2' = M_1 L_2 \tilde{M}_1$ with $M_1 = cos(\theta_{tilt}) + sen(\theta_{tilt}) L_1$. In this experiment, a point on the circuit board was selected, and, using the KLT algorithm, the displacement was tracked in the image transforming the velocities of the pan-tilt's joint using the visual-mechanical Jacobian of Eq. VI-D.

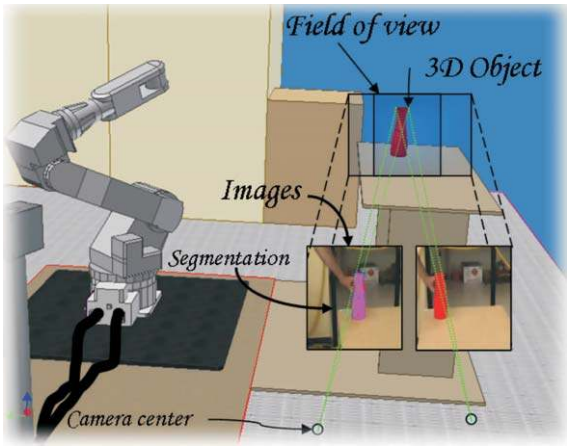

Fig. 12. Mathematical model of the object.

In Fig. 11, a sequence of images captured by the robot can be seen. In these images, one can see that the position of the circuit board does not change while the background is in continuous change.

Fig. 13. Pose estimation of a disc with a fixed camera.

Fig. 14. Pose estimation of a recipient.

7. Pose estimation

There are many approaches to solve the pose estimation problem. This particular approach, the known mathematical model of the object is projected on the camera's image. This is possible because, after the calibration, the intrinsic parameters of the camera are known, see Fig. 12. The image of the mathematical projected model is compared with the image of the segmented object. If a match is found between them, then this means that the mathematical object is placed in the same position and orientation than the real object. Otherwise, the descendant gradient will be followed to rotate and translate the mathematical model, in order to reduce the registration error.

Figure 13 shows the pose estimation result. In this case, there is a maximum error of 0.4° in the orientation estimation and 5 mm of maximum error in the position estimation of the object. The problem becomes more difficult to solve when the stereoscopic system is moving. Figure 14 shows the result of this approach. To know the real object's position with respect to the world coordinate system, it is necessary to know the extrinsic camera's parameters. Figure 15 illustrates the object's position and orientation with respect to the robot's hand. In the upper row of this image, an augmented reality position sequence of the object can be seen. This means that the mathematical object can be added in the real image. Furthermore, in the second row of the same image, the result of the pose estimation can be seen in virtual reality.

8. Grasping the objects

The complexity of grasping objects depends greatly on the nature of the object surface, its weight, and its 3D location. Platonic objects are easy to recognize and grasp. In contrast, real objects can have high nonlinear surfaces, holes, and can even be flexible. This paper follows a simple approach based on one of Newton's basic rules, which takes into consideration certain forces and moments in equilibrium. The robot grasper has to identify key grasping points. In this regard, grasping rules will be developed using geometric products of entities like points, lines, planes, circles, and spheres in order to discover the key grasping points. Due to the simplicity of the rules, they are not computational time consuming.

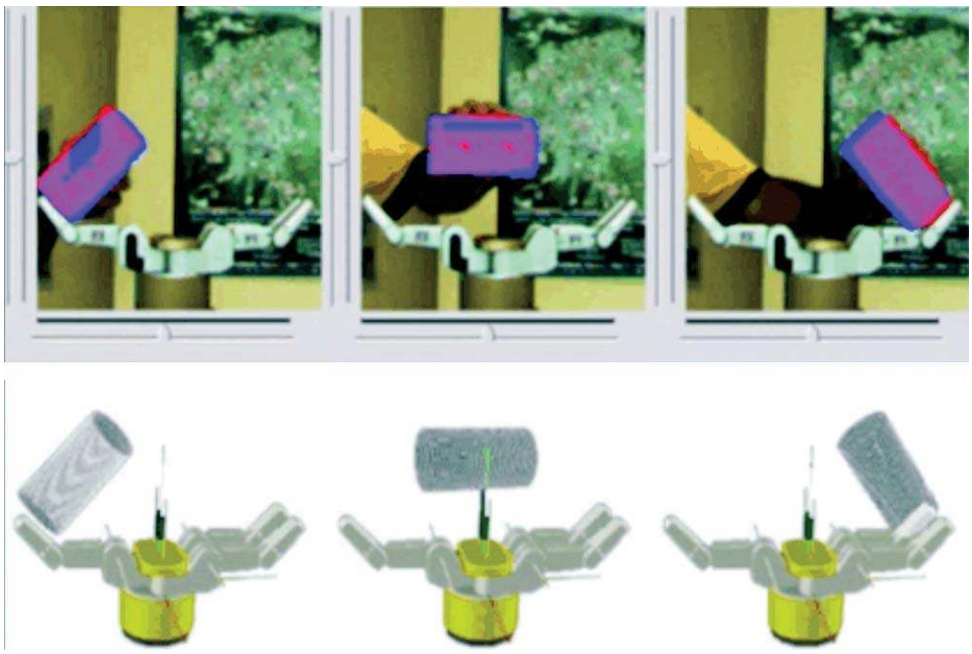

Fig. 15. Augmented reality representing an object and the robot hand.

Table 2
Functions describing the surface of objects

Particle	$H = 3e_1 + 4e_2 + 5e_3$
Sphere	$H = cos(s)cos(t)e_1 + cos(s)sin(t)e_2 + sin(s)e_3$
Cylinder	$H = cos(t)e_1 + sin(t)e_2 + se_3$
Plane	$H = te_1 + se_2 + (3s + 4t + 2)e_3$

Considering that the use of cameras only allows vision of the surface of the observed objects, this paper will consider these surfaces as bi-dimensional and embedded in a 3D space. The surfaces can be described by the following function

$$H(s,t) = h_x(s,t)e_1 + h_y(s,t)e_2 + h_z(s,t)e_3 \qquad (88)$$

where s and t are real parameters in the range [0, 1]. Such parametrization allows us to work with different objects like points, conics, quadric, or even more complex objects like cups, glasses, etc.

Since the objective is to grasp such objects with the Barrett Hand, the consideration that it only has three fingers must be made. Therefore, the problem consists in finding three "touching points" for which the system is in equilibrium during the grasping; this means that the sum of the forces equals to zero, as well as the sum of the moments. For this case, the consideration

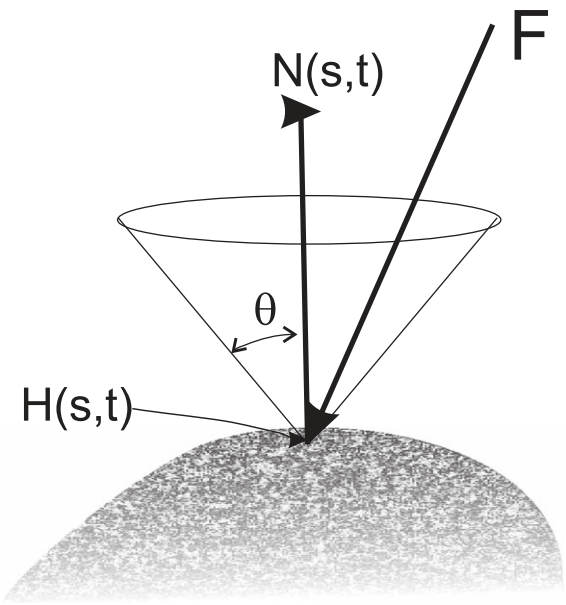

Fig. 16. The friction cone.

that there friction exists in each "touching point" was made.

As friction is being considered, a set of forces that can be applied exist over surface $H(s,t)$; such forces

are inside a cone which has a normal vector $N(s,t)$ of the surface as its axis (as shown in Fig. 16), and its radius depends on the friction's coefficient.

$$\|F - F_n\| \leq -\mu(|F_n|), \tag{89}$$

$F_n = (F \cdot N(s,t))N(s,t)$ is the normal component of F. The angle for the incidence of F with respect to the normal can be calculated using the wedge product and should be smaller than a fixed θ_μ

$$\frac{\|F \wedge N(s,t)\|}{F \cdot N(s,t)} \leq tan(\theta_\mu). \tag{90}$$

The surface of the object is known, so its normal vector can be computed in each point using

$$N(s,t) = \left(\frac{\partial H(s,t)}{\partial s} \wedge \frac{\partial H(s,t)}{\partial t}\right) I_e. \tag{91}$$

In surfaces with little friction, angle θ is very small. Therefore, the value of F tends to its projection over the normal ($F \approx F_n$). To maintain equilibrium, the sum of the forces must be zero (Fig. 17).

$$\sum_{i=1}^{3} \|F_n\| N(s_i, t_i) = 0. \tag{92}$$

This fact restricts the grasp points over the surface in which the forces can be applied. This number of points is more reduced if it's considered that the forces over the object are equal considering the unit normal.

$$\sum_{i=1}^{3} N(s_i, t_i) = 0. \tag{93}$$

Additionally, to maintain the equilibrium, the sum of moments must be zero

$$\sum_{i=1}^{3} H(s,t) \wedge N(s,t) = 0. \tag{94}$$

The points on the surface having the same directed distance to the center of mass of the object fulfill $H(s,t) \wedge N(s,t) = 0$. Because the normal in such points crosses the center of mass (C_m), it does not produce any moment. Before determining the external and internal points, the center of mass must be computed as

$$C_m = \int_0^1 \int_0^1 H(s,t) ds dt. \tag{95}$$

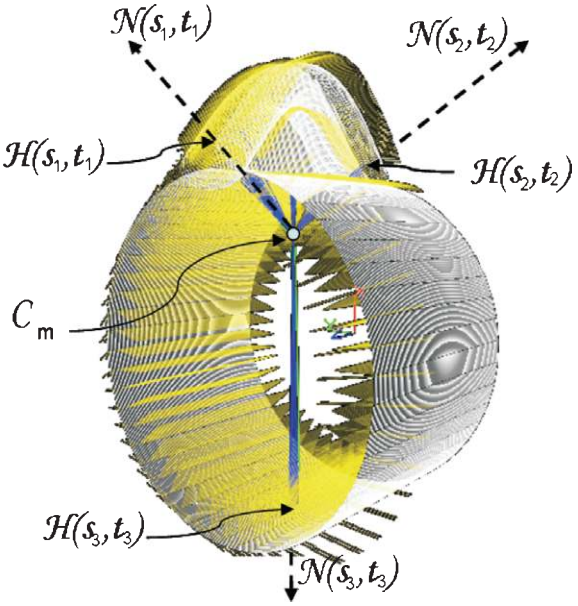

Fig. 17. Object and its normal vectors.

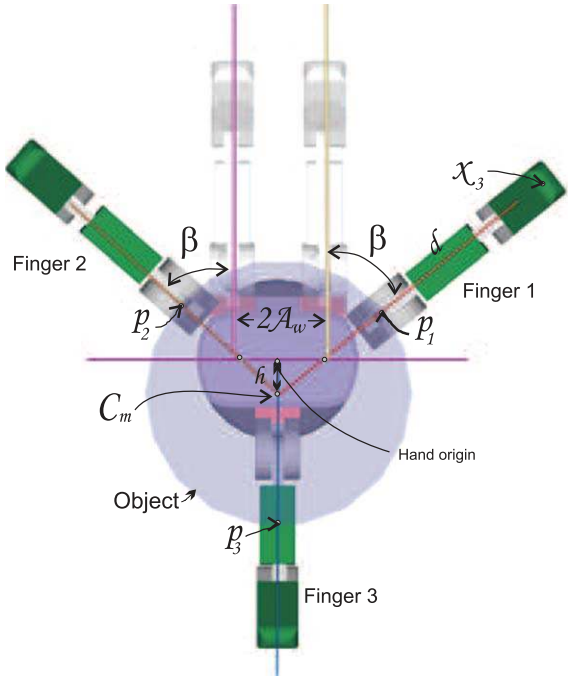

Fig. 18. Object relative position.

Once C_m is calculated, the next constraint can be established

$$(H(s,t) - C_m) \wedge N(s,t) = 0. \tag{96}$$

The values s and t satisfying (96) form a subspace called "grasping space", and they accomplish that $H(s,t)$ are critical points on the surface (being maximums, minimums, or inflections). The equation (96) is hard to fulfill due to the noise, and it is necessary to consider a cone of vectors. So, an angle called α is introduced,

$$\frac{\|(H(s,t) - C_m) \wedge N(s,t)\|}{(H(s,t) - C_m) \cdot N(s,t)} \leq tan(\alpha) \tag{97}$$

Since cameras are being manipulated, a cloud of points can be obtained, and a surface can be interpolated between these points. This procedure, however, introduces errors and equation (96), unfortunately, cannot be satisfied; for this reason, equation (97) is used, which allows some looseness.

The constraint imposing that the three forces must be equal is hard to fulfill because it implies that the three points must be symmetric with respect to the mass center. When such points are not present, the constraint can be relaxed to allow that only two forces are equal in order to fulfill the hand's kinematic equations. Then, the normals $N(s_1, t_1)$ and $N(s_2, t_2)$ must be symmetric with respect to $N(s_3, t_3)$ (see Fig. 18).

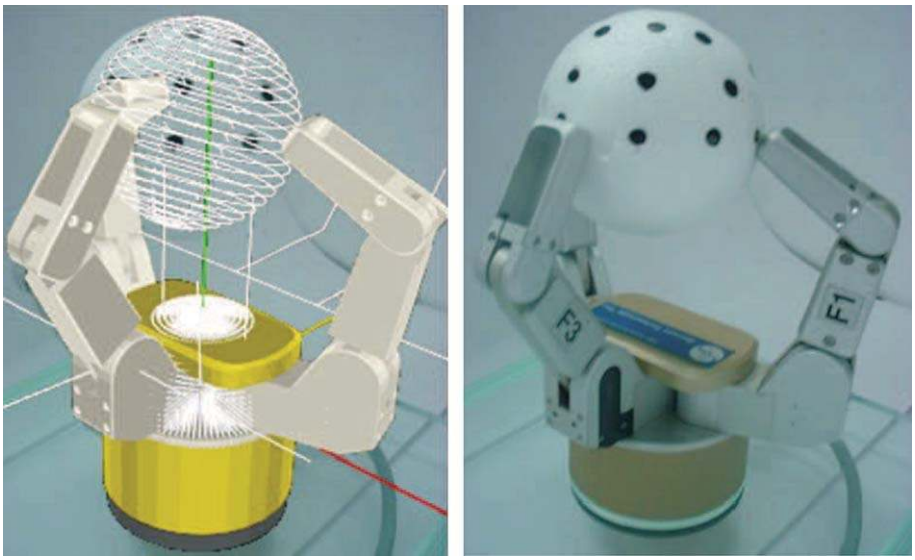

Fig. 19. Grasping an sphere.

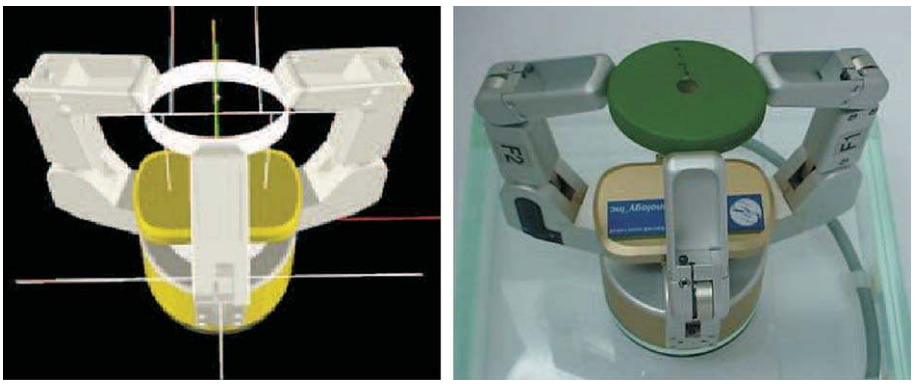

Fig. 20. Grasping a disc.

Fig. 21. Grasping an egg.

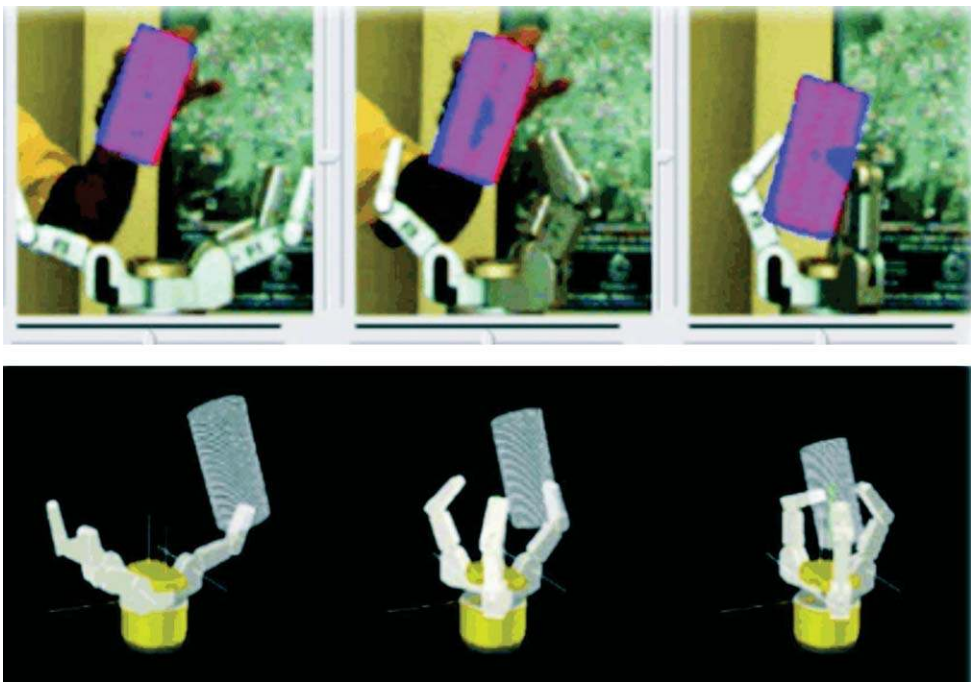

Fig. 22. Visually guided grasping.

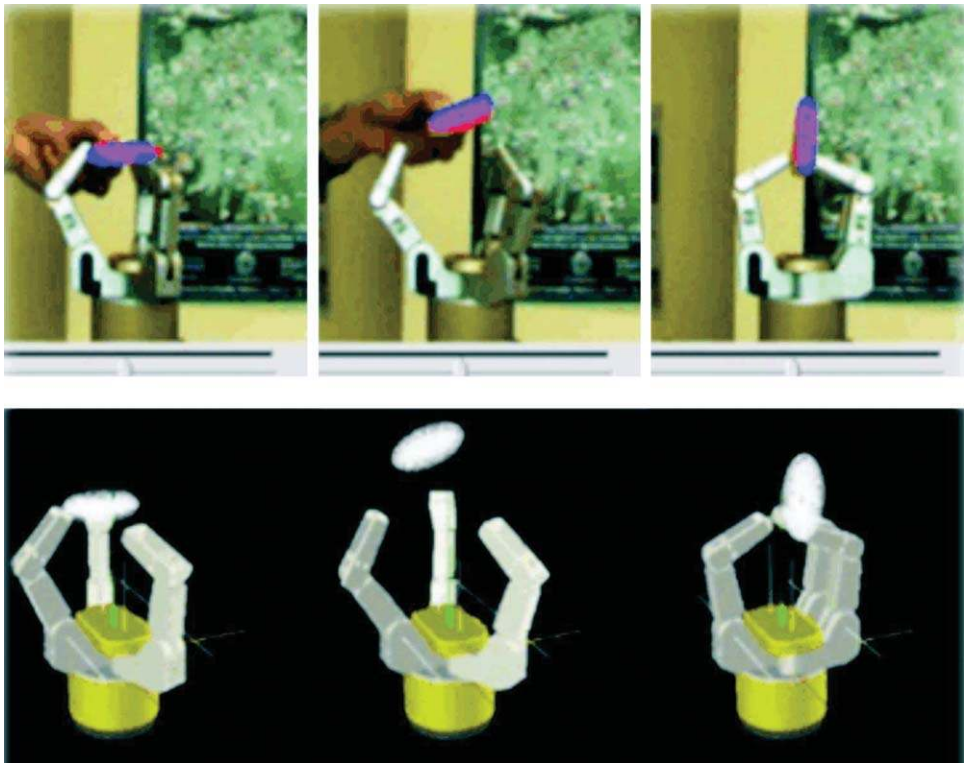

Fig. 23. Changing the object's pose.

$$N(s_3, t_3)N(s_1, t_1)N(s_3, t_3)^{-1} = N(s_2, t_2) \quad (98)$$

Once the three grasping points ($P_1 = H(s_1, t_1)$, $P_2 = H(s_2, t_2)$, $P_3 = H(s_3, t_3)$) are calculated, it is really easy to determine the angles at the joints of each finger. To determine the angle of the spread ($q_4 = \beta$), the following are used

$$cos(\beta) = \frac{(p_1 - C_m) \cdot (C_m - p_3)}{|p_1 - c_m|\,|C_m - p_3|} \quad (99)$$

$$sin(\beta) = \frac{|(p_1 - C_m) \wedge (C_m - p_3)|}{|p_1 - c_m|\,|C_m - p_3|} \quad (100)$$

To calculate each one of the finger angles, its elongation is determined as

$$x_3' \cdot e_2 = |(p_1 - C_m)| - \frac{A_w}{\sin(\beta)} - A_1. \quad (101)$$

$$x_3' \cdot e_2 = |(p_2 - C_m)| - \frac{A_w}{\sin(\beta)} - A_1. \quad (102)$$

$$x_3' \cdot e_2 = |(p_3 - C_m)| + h - A_1. \quad (103)$$

where $x_3' \cdot e_2$ determines the opening distance of the finger

$$x_3' \cdot e_2 = (M_2 M_3 x_3 \widetilde{M}_3 \widetilde{M}_2) \cdot e_2 \quad (104)$$

$$x_3' \cdot e_2 = A_1 + A_2 \cos\left(\tfrac{1}{125} q + I_2\right) + A_3 \cos\left(\tfrac{4}{375} q + I_2 + I_3\right) \quad (105)$$

Solving for angle q results in the opening angle for each finger. On the other hand, transformation M must be found, which allows the hand to be put in a such way that each finger-end coincides with the corresponding contact-point. Such transformation M is divided in three transformations (M_1, M_2, M_3) for the sole purpose of an easy explanation. With the same purpose, the finger-ends are labeled as X_1, X_2, and X_3, and the contact-points as P_1, P_2, and P_3.

The first transformation M_1 is the translation between the object and the hand, which is equal to the directed distance between the centers of the circles called $Z_h^* = X_1 \wedge X_2 \wedge X_3$ and $Z_o^* = P_1 \wedge P_2 \wedge P_3$, and it can be calculated as

$$M_1 = e^{-\frac{1}{2}\left(\frac{Z_h^*}{Z_h^* \wedge e_\infty} \wedge \frac{Z_o^*}{Z_o^* \wedge e_\infty} \wedge e_\infty\right) I_c} \tag{106}$$

The second transformation allows the alignment of the planes $\pi_h^* = Z_h^* = X_1 \wedge X_2 \wedge X_3 \wedge e_\infty$ and $\pi_o^* = Z_o^* \wedge e_\infty$, which are generated by the new points of the hand and the object. This transformation is calculated as

$$M_2 = e^{-\frac{1}{2}\pi_h \wedge \pi_o} \tag{107}$$

The third transformation allows the points to overlap. This can be calculated using the planes $\pi_1^* = Z_o \wedge X_3 \wedge e_\infty$ and $\pi_2^* = Z_o \wedge P_3 \wedge e_\infty$, which are generated by the circle's axis and any of the points

$$M_3 = e^{-\frac{1}{2}\pi_1 \wedge \pi_2} \tag{108}$$

The transformation $M = M_3 M_2 M_1$ can be parametrized to generate a trajectory for the object grasping.

9. Results

This section presents the experimental results of our grasping algorithm. Each pair of images corresponds to the simulated and the real one (see images 15–18). These images show the Barret hand in action. For each object, the algorithm manages to find the optimal grasp points, so that the object is held properly and in equilibrium. Note that the found points correspond to the natural grasping points.

The results of the combination of pose estimation algorithms, visual control, and grasping are presented to create a new algorithm for visually guided grasping. In image 22 a sequence of images of the grasping achieved is presented. As the bottle approaches the hand, the fingers look for possible grasp points.

When the object pose or the object itself is changed, the algorithm computes a new grasping approach. Image (23) shows a sequence of images changing the pose of the object.

10. Conclusion

The visual Jacobian matrix and direct kinematics of robotic devices such as the Barrett hand and the Pan-Tilt unit were modeled using the powerful conformal geometric algebra as computational framework. To close the loop between perception and action, the poses of the object and the hand are estimated. Then, a control law for approaching and grasping is applied. This control law is geometrically formulated using the visual-mechanical Jacobian matrix, which in turn is computed using the principal lines of the camera and the axis of the pan-tilt unit. This paper proves that based on the intrinsic information of an object, it is possible to find a feasible grasping strategy. In this case, the conformal geometric algebra helps resolve this kind of problems using lines and planes as vector or bivectors instead of points as it is done in classical vector calculus. The experimental results confirm the efficiency of these algorithms for visual tracking, grasping, pose estimation, and visually guided grasping.

References

[1] H. Li, D. Hestenes and A. Rockwood, Generalized homogeneous coordinates for computational geometry, in *Geometric Computing with Clifford Algebras*, G. Somer, ed., Springer-Verlag, Heidelberg, 2001, pp. 27–52.

[2] E. Bayro-Corrochano and D. Kähler, Motor algebra approach for computing the kinematics of robot manipulators, *Journal of Robotics Systems* **17**(9) (2000), 495–516.

[3] E. Bayro-Corrochano, Robot perception using clifford conformal geometry, Chap. 13, in *Handbook of Computational Geometry for Pattern Recognition, Computer Vision, Neurocomputing and Robotics*, E. Bayro-Corrochano, ed., Springer Verlag, Heidelberg, May 2005.

[4] L.E. Falcon-Morales and E. Bayro-Corrochano, Design of algorithms of robor vision using conformal geometric algebra. International Mathematical Forum, *Journal of Theory and Applications* **2** (2007), 17–20.