Self calibration method of binocular vision based on Conformal geometric algebra

Stanislav Frolík[1], Marek Stodola[1]

[1] Institute of Mathematics, Faculty of Mechanical Engineering

Brno University of Technology Brno Czech Republic

{Marek.Stodola, Stanislav.Frolik } @vutbr.cz

Abstract

We will study binocular vision for 6-DOF robotic manipulator in conformal geometric algebra approach. We will focus on the case where some information as relative cameras positions, has been lost. In particular, we will use the construction of the manipulator to infer a self calibration method for cameras position based in binocular vision with incomplete information.

AMS2020— Primary 15A66; Secondary 51N25

Keywords— Conformal geometric algebra, Binocular vision, self calibration, robotic manipulator

The research was supported by the Grant No.: FSI-S-20-6187

1 Introduction

Computer vision has been here for a while, but since the spread of neural networks and artificial intelligence models in the past decade it became trendy and gained a lot of popularity in research and industry as well. With wide range of applications - from facial recognition, through industrial measurements to modern microscopy - researchers and developers pushed hardware to the limit. Moreover, just one sensor can not secure satisfactory field of view, reliability and stereopsis. To ensure such conditions, one has to use binocular vision. In order to achieve a solid reconstructed scene, one has to ensure either fixed relative position of sensors, which is due to material flexibility always a problem, or compute the position from some reference object, which is operationally hard to achieve since the object has to be fixed in predefined configuration (pose). In this paper we would like to propose a method which resolves such troubles. Even though there are another work-arounding methods, e.g. gradient based and sometimes machine learning, we are intended to proceed in more geometric way. We will investigate a case where we will reconstruct 3D position of observed object, given by images from two cameras. Our goal is to solve a case where we have lost cameras positions, which, in general, has to be known for further computation, i.e. calibrate cameras position, which leads to 3D scene reconstruction. Moreover our approach does camera self-calibration, thus cameras (or binocular vision) does not require any special calibration method. In order to get the 3D position of observed object we will use a minimum knowledge of observed object construction.

This approach will be demonstrated with a typical example of robotic manipulator. For this problem we will use Conformal geometric algebra. At first we will recall some notions such as multivectors, transformators etc. from CGA in section 2. Then, in section 3 we will infer equations for determination of manipulator position given by cameras position and its images. Those equations are useful for further computations. Inversely we will define equations for camera images given by cameras and manipulator position. With such knowledge we can solve a problem of binocular vision in the case where some information (e.g. camera position) has been lost in section 4. It means that we do not have any information of cameras position neither manipulator position. Traditional task of binocular vision is to compute position of object from cameras position or vice-versa. This is not the case. Fortunately we have some knowledge of observed object like kinematics, dimensions, specific construction etc. which can by used to infer the unknown positions.

2 Conformal geometric algebra

We will infer further computations using geometric algebra approach, which can be found in classical books [15, 4, 5, 2, 12] or scientific papers (e.g. [7, 9]) published in last decade. In this chapter we will recall only some necessary notions from the theory of geometric algebra calculus. We will recall just statements and formulas which will be strictly needed. For compact theory see the books listed above.

In the binocular vision and robotics, the conformal geometric algebra (CGA) plays significant role. So, let us have conformal geometric algebra, i.e. geometric algebra $\mathbb{G}_{4,1}$ together with Hestenes embedding

$$(x,y,z) \in \mathbb{R}^3 \mapsto xe_1 + ye_2 + ze_3 + \frac{1}{2}(xe_1 + ye_2 + ze_3)^2 e_\infty + e_0. \tag{1}$$

Generally in Geometric algebra models we represent objects by so called OPNS and IPNS representations with the help of dot and wedge product. In detail, for each point $\mathbf{P}$ and each geometric object $\mathbf{O}$ in IPNS representation, resp. geometric object $\mathbf{O}^*$ in OPNS representation the following property holds:

$$\mathbf{P} \in \mathbf{O} \quad \Leftrightarrow \quad \mathbf{P} \cdot \mathbf{O} = 0, \tag{2}$$

$$\mathbf{P} \in \mathbf{O}^* \quad \Leftrightarrow \quad \mathbf{P} \wedge \mathbf{O}^* = 0. \tag{3}$$

Follows relation between IPNS and OPNS representation of object $\mathbf{O}$:

$$\mathbf{O} = cI\mathbf{O}^*, \quad I = e_1 \wedge e_2 \wedge e_3 \wedge e_\infty \wedge e_0, \quad c \in \mathbb{R} \setminus \{0\}. \tag{4}$$

It allows us to represent geometric objects in forms as in Table 1.

Let us recall some properties of conformal geometric algebra. During the following algorithms of binocular vision these relationships between geometric objects are used:

Object	IPNS representation	OPNS representation
Point	$\mathbf{P} = \mathbf{p} + \frac{1}{2}\|\mathbf{p}\|^2 e_\infty + e_0$	
Sphere	$\mathbf{S} = \mathbf{P} - \frac{1}{2}r^2 e_\infty$	$\mathbf{S}^* = \mathbf{P}_1 \wedge \mathbf{P}_2 \wedge \mathbf{P}_3 \wedge \mathbf{P}_4$
Plane	$\pi = \mathbf{n} + d e_\infty$	$\pi^* = \mathbf{P}_1 \wedge \mathbf{P}_2 \wedge \mathbf{P}_3 \wedge e_\infty$
Circle	$\mathbf{Z} = \mathbf{S}_1 \wedge \mathbf{S}_2$	$\mathbf{Z}^* = \mathbf{P}_1 \wedge \mathbf{P}_2 \wedge \mathbf{P}_3$
Line	$\mathbf{L} = \pi_1 \wedge \pi_2$	$\mathbf{L}^* = \mathbf{P}_1 \wedge \mathbf{P}_2 \wedge e_\infty$
Point pair	$\mathbf{P}_p = \mathbf{S}_1 \wedge \mathbf{S}_2 \wedge \mathbf{S}_3$	$\mathbf{P}_p^* = \mathbf{P}_1 \wedge \mathbf{P}_2$
	$\mathbf{p} = p_1 e_1 + p_2 e_2 + p_3 e_3,$	$\mathbf{n} = n_1 e_1 + n_2 e_2 + n_3 e_3, \quad \|\|\mathbf{n}\|\| = 1$

Table 1: Geometric objects in CGA

1. Distance between two points: $d_{\mathbf{P}_1\mathbf{P}_2} = \sqrt{-2(\mathbf{P}_1 \cdot \mathbf{P}_2)}$
2. Angle between two planes: $\cos\alpha = \pi_1 \cdot \pi_2$
3. Plane of symmetry of two points:
$$\pi = \mathbf{P}_1 - \mathbf{P}_2 \tag{5}$$
4. Midpoint $\mathbf{M}$ of point pair $\mathbf{P}_1 \wedge \mathbf{P}_2$ as point pair with e_∞:
$$c_1(\mathbf{M} \wedge e_\infty) = (c(\mathbf{P}_1 \wedge \mathbf{P}_2) \cdot e_\infty) \cdot \mathbf{L}_{\mathbf{P}_1\mathbf{P}_2}, \quad c, c_1 \in \mathbb{R}/\{0\}, \tag{6}$$
where
$$\mathbf{L}_{\mathbf{P}_1\mathbf{P}_2} = I(c(\mathbf{P}_1 \wedge \mathbf{P}_2) \wedge e_\infty). \tag{7}$$
5. Distance between points of point pair $\mathbf{P}_1 \wedge \mathbf{P}_2$:
$$d_{\mathbf{P}_1\mathbf{P}_2} = 2\sqrt{\frac{c(\mathbf{P}_1 \wedge \mathbf{P}_2) \cdot c(\mathbf{P}_1 \wedge \mathbf{P}_2)}{I\mathbf{L}_{\mathbf{P}_1\mathbf{P}_2} \cdot I\mathbf{L}_{\mathbf{P}_1\mathbf{P}_2}}} \tag{8}$$

For the following computations, let us introduce a couple of new operators, namely $\bullet$ and $\circ$ as follows:

$$\bullet\pi = \frac{\pi}{\sqrt{\pi \cdot \pi}}, \tag{9}$$
$$\circ\mathbf{L} = \frac{\mathbf{L}}{\sqrt{-\mathbf{L} \cdot \mathbf{L}}}, \tag{10}$$

then $\bullet\pi$ is a standard representation of plane π and $\circ\mathbf{L}$ is a standard representation of line $\mathbf{L}$. In standard representation, a plane, resp. a line has unit normal vector, resp. unit direction vector. In the end, remind that by translator we mean a multivector

$$T = \exp(-\frac{1}{2}\mathbf{t}e_\infty) = \sum_{i=0}^{\infty} \frac{(-\frac{1}{2}\mathbf{t}e_\infty)^i}{i!} = 1 - \frac{1}{2}\mathbf{t}e_\infty, \tag{11}$$

where $t = xe_1 + ye_2 + ze_3$ is a translate vector with its conjugate

$$\tilde{T} = \exp(\frac{1}{2}\mathbf{t}e_\infty) = 1 + \frac{1}{2}\mathbf{t}e_\infty. \tag{12}$$

By rotor we mean a multivector

$$R = \exp(-\frac{\theta}{2}\mathbf{L}_{st}), \tag{13}$$

where $\mathbf{L}_{st}$ is standard IPNS representation of rotation axis and $\theta \in \mathbb{R}$ is a rotation angle, with its conjugate

$$\tilde{R} = \exp(\frac{\theta}{2}\mathbf{L}_{st}). \tag{14}$$

Recall that by motor we mean a rigid body motion

$$M = RT \tag{15}$$

with its conjugate

$$\tilde{M} = \tilde{T}\tilde{R}. \tag{16}$$

Remind that having a geometric object $\mathbf{O}$ we compute rigid body motion with help of conjugation by motor M, e.g. $\tilde{\mathbf{O}} = M\mathbf{O}\tilde{M}$.

3 3D scene reconstruction

There are various concepts of 3D scene reconstruction. We focus on so called binocular vision, for example see [8, 14, 13] (for monocular vision see [16] for multiocular vision see [1]). At first, we define forward kinematics in CGA, this became a standard approach, see [10, 18, 19, 3, 6] (for inverse kinematic in geometric algebra approach see [17, 6, 11]).

3.1 Forward kinematics

The goal of this subsection is to compute the final position of the manipulator after applying rotational motion in its joints. Let us denote angles of rotations as

$$\theta_{01}, \theta_{12}, \theta_{34}, \theta_{56}, \theta_{67}, \theta_{78}.$$

The final position is given by points $\tilde{P}_i \in \{0, 1, \ldots, 8\}$, where $\tilde{\mathbf{P}}_i$ represents point $\mathbf{P}_i$ after a rotation.

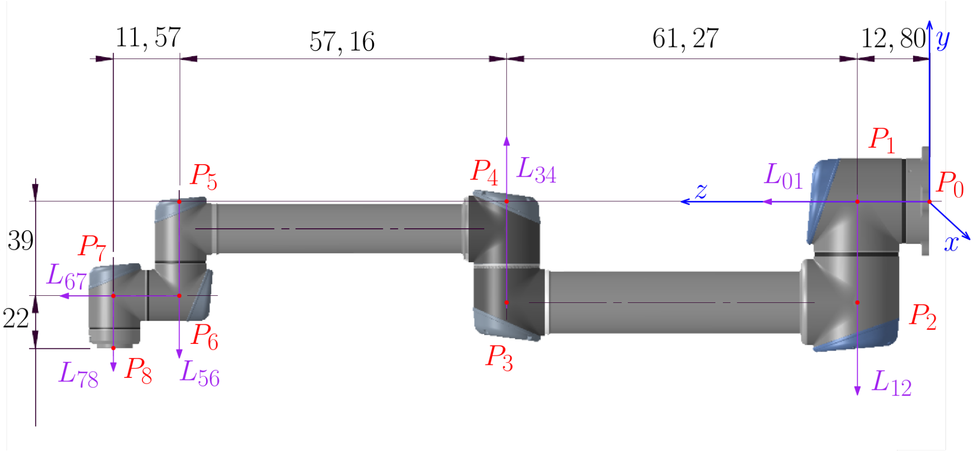


Figure 1: An example of robotic manipulator. Blue lines represent coordinate system axes, purple lines represent rotation axes, red points describe robot position.

Obviously point P_0 is identified as origin of our coordinate system and point P_1 never change its position, thus $\tilde{P}_0 = P_0$ and $\tilde{P}_1 = P_1$. Now let us denote axes of rotation corresponding to θ_{ij} as $\tilde{L}_{ij}$ and corresponding rotors as R_{ij}. Then we can compute

$$\begin{aligned} \tilde{\mathbf{L}}^*_{ij} &= \circ(\tilde{\mathbf{P}}_i \wedge \tilde{\mathbf{P}}_j \wedge e_\infty), && (17) \\ R_{ij} &= \exp(-\frac{\theta_{ij}}{2}\tilde{\mathbf{L}}_{ij}). && (18) \end{aligned}$$

Finally, we can compute the final positions of points $\tilde{P}_i$ after applying a transformation as

$$\begin{aligned} &\tilde{\mathbf{P}}_2 = R_{01}\mathbf{P}_2\tilde{R}_{01}, \\ &\tilde{\mathbf{P}}_3 = R_{12}R_{01}\mathbf{P}_3\tilde{R}_{01}\tilde{R}_{12}, \\ &\quad\vdots \\ &\tilde{\mathbf{P}}_8 = R_{67}R_{56}R_{34}R_{12}R_{01}\mathbf{P}_8\tilde{R}_{01}\tilde{R}_{12}\tilde{R}_{34}\tilde{R}_{56}\tilde{R}_{67}. \end{aligned} \qquad (19)$$

3.2 Binocular vision

In this subsection we will determine a position of the manipulator by an image from two cameras. Input parameters of such problem are positions of left camera focus $\mathbf{O}_L = [o_{L1}, o_{L2}, o_{L3}] \equiv o_{L1}e_1 + o_{L2}e_2 + o_{L3}e_3 + \frac{1}{2}\|\mathbf{o}_L\|^2 + e_0$ and right camera focus respectively $\mathbf{O}_R = [o_{R1}, o_{R2}, o_{R3}] \equiv o_{R1}e_1 + o_{R2}e_2 + o_{R3}e_3 + \frac{1}{2}\|\mathbf{o}_R\|^2 + e_0$. Parameters $\alpha_L, \beta_L, \gamma_L$ describe the view angle of left camera, where α_L describes a rotation around vertical axis, β_L describes a rotation around horizontal axis and γ_L describes a rotation around anteroposterior axis. Analogically we assign to $\alpha_R, \beta_R, \gamma_R$ corresponding angles for the right camera. For simplicity, let us assume that those reference axes go through the focuses of cameras. Parameters f_L, f_R represents focal distance of cameras. For simplicity assume that they are equal $f_L = f_R = f$. π_{prL}, π_{prR} denotes projection planes and images of points $\tilde{\mathbf{P}}_i, i \in \{0, 1, 2, \ldots, 8\}$ we denote as $\mathbf{P}_{iL}$ resp. $\mathbf{P}_{iR}$.

Firstly, let us place camera focuses and images of projection planes into the space. Initially we place images of projection planes into plane $\pi_{yz} = e_1$ and camera focuses to point $\mathbf{O} = fe_1 + \frac{1}{2}f^2e_\infty + e_0$. Now we need to determine motor $M_L = T_LR_L$ given by parameters $o_{L1}, o_{L2}, o_{L3}, f, \alpha_L, \beta_L, \gamma_L$, which realize such geometric transformation which grants appropriate position of left camera focus and images from left projection plane in the space. Equivalently we determine motor $M_R = T_LR_L$ given by $o_{R1}, o_{R2}, o_{R3}, f, \alpha_R, \beta_R, \gamma_R$.

Remark 3.1 *During computation of 3D scene reconstruction we assume that rotations R_L, R_R of cameras are given. In fact, a technician can set cameras with angles $\alpha_R, \beta_R, \gamma_R$, resp. $\alpha_L, \beta_L, \gamma_L$. Therefore we infer rotations of cameras with those parameters in this article.*

We can compute axis z as

$$\mathbf{L}_z = e_1 \wedge e_2. \qquad (20)$$

Now we can move this axis to point $\mathbf{O}$ with translator $T = 1 - \frac{1}{2}\mathbf{t}e_\infty$, where $\mathbf{t} = fe_1$, and convert it to standard form:

$$\mathbf{L}_{\overline{z}} = T\mathbf{L}_z\tilde{T}. \tag{21}$$

Furthermore we will define a rotor which represents rotation of left camera around vertical axis $\mathbf{L}_{\overline{z}}$ by angle α_L in negative direction.

$$R_{z_L} = \exp(\frac{\alpha_L}{2}\mathbf{L}_z). \tag{22}$$

Now we will move line $\mathbf{L}_y = e_3 \wedge e_1$ with translator T and we will rotate it with rotor R_{z_L}, to preserve horizontal axis fixed with camera

$$\mathbf{L}_{\overline{y}} = T\mathbf{L}_y\tilde{T}, \tag{23}$$

$$\mathbf{L}_{\tilde{\overline{y}}_L} = R_{z_L}\mathbf{L}_{\overline{y}}\tilde{R}_{z_L}. \tag{24}$$

Now we will define rotors representing rotation around horizontal axis $\mathbf{L}_{\tilde{\overline{y}}_L}$ by angle β_L in negative direction:

$$R_{y_L} = \exp(\frac{\beta_L}{2}\mathbf{L}_{\tilde{\overline{y}}_L}). \tag{25}$$

Next we will rotate axis $\mathbf{L}_x = e_2 \wedge e_3$ with rotors R_{z_L} and R_{y_L}. After that we will compute rotor representing rotation around this anteroposterior axis of left camera by angle γ_L, in negative direction.

$$\mathbf{L}_{\tilde{\tilde{x}}_L} = R_{y_L}R_{z_L}\mathbf{L}_x\tilde{R}_{z_L}\tilde{R}_{y_L}, \tag{26}$$

$$R_{x_L} = \exp(\frac{\gamma_L}{2}\mathbf{L}_{\tilde{\tilde{x}}_L}). \tag{27}$$

Finally we will define translator representing translation to the appropriate position

$$\mathbf{t}_L = (o_{L1} - f)e_1 + o_{L2}e_2 + o_{L3}e_3, \tag{28}$$

$$T_L = 1 - \frac{1}{2}\mathbf{t}_L e_\infty. \tag{29}$$

Resulting in motor M_L

$$M_L = T_L R_{x_L} R_{y_L} R_{z_L}, \tag{30}$$

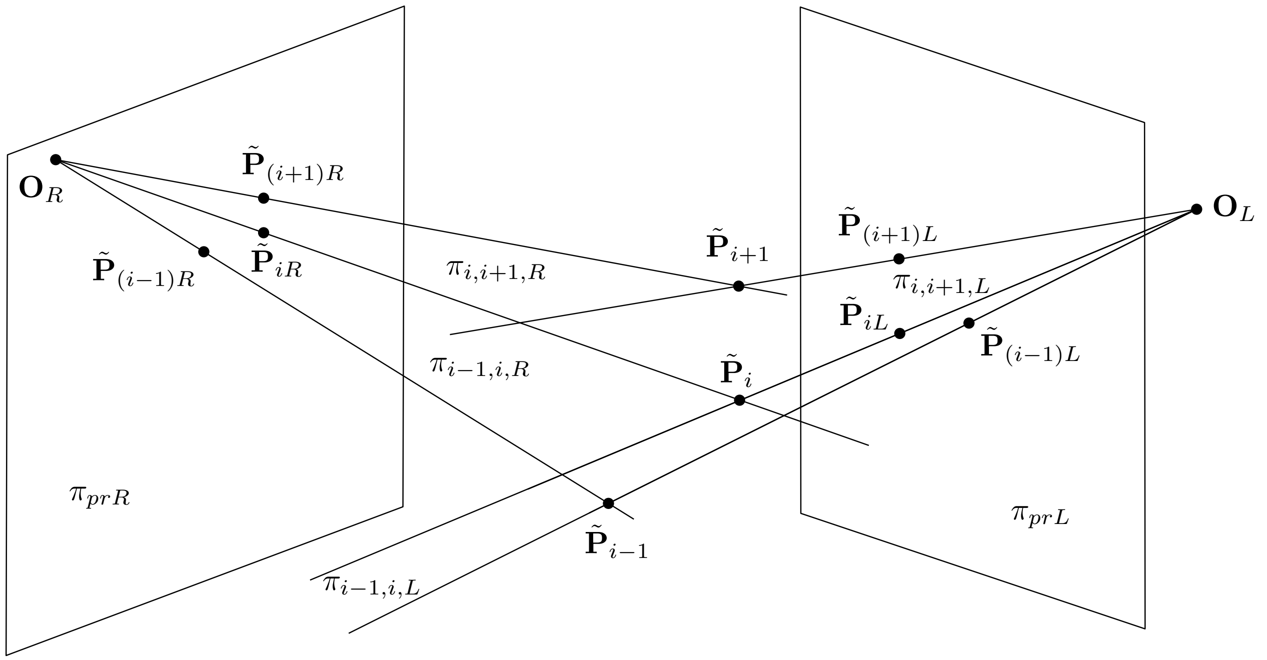
and its conjugate motor

$$\tilde{M}_L = \tilde{R}_{z_L}\tilde{R}_{y_L}\tilde{R}_{x_L}\tilde{T}_L. \tag{31}$$

Analogically we can get motor M_R and its conjugate rotor $\tilde{M}_R$ for the right camera. We will place image records into space with motors M_L and M_R as

$$\tilde{\mathbf{P}}_{iL} = M_L\mathbf{P}_{iL}\tilde{M}_L, \tag{32}$$

$$\tilde{\mathbf{P}}_{iR} = M_R\mathbf{P}_{iR}\tilde{M}_R. \tag{33}$$



$\mathbf{O}_R$
$\tilde{\mathbf{P}}_{(i+1)R}$
$\tilde{\mathbf{P}}_{iR}$
$\tilde{\mathbf{P}}_{(i-1)R}$
$\pi_{i,i+1,R}$
$\pi_{i-1,i,R}$
$\tilde{\mathbf{P}}_{i+1}$
$\tilde{\mathbf{P}}_{(i+1)L}$
$\mathbf{O}_L$
$\pi_{i,i+1,L}$
$\tilde{\mathbf{P}}_{iL}$
$\tilde{\mathbf{P}}_{(i-1)L}$
$\tilde{\mathbf{P}}_i$
π_{prR}
π_{prL}
$\tilde{\mathbf{P}}_{i-1}$
$\pi_{i-1,i,L}$

Figure 2: Position reconstruction of points $\tilde{\mathbf{P}}_i$ in space

Now we will introduce planes

$$\begin{aligned} \pi_{i,i+1,L} &= I(\mathbf{O}_L \wedge \tilde{\mathbf{P}}_{iL} \wedge \tilde{\mathbf{P}}_{(i+1)L} \wedge e_\infty), && (34)\\ \pi_{i,i+1,R} &= I(\mathbf{O}_R \wedge \tilde{\mathbf{P}}_{iR} \wedge \tilde{\mathbf{P}}_{(i+1)R} \wedge e_\infty), \quad i \in \{0,1,...,7\}. && (35)\end{aligned}$$

Point $\tilde{\mathbf{P}}_i$, $i \in \{1,2,...,7\}$ is an intersection of planes from set $\Pi_i = \{\pi_{i-1,i,L}, \pi_{i,i+1,L}, \pi_{i-1,i,R}, \pi_{i,i+1,R}\}$ (see figure 2). Conformal geometric algebra allows to find an intersection of three planes as a pair of points, where one of the point is e_∞.

$$c(\tilde{\mathbf{P}}_i \wedge e_\infty) = I(\pi_{i1} \wedge \pi_{i2} \wedge \pi_{i3}), \tag{36}$$

where

$$c \in \mathbb{R} \setminus \{0\},\ \pi_{i1}, \pi_{i2}, \pi_{i3} \in \Pi_i, \pi_{ij} \wedge \pi_{ik} \neq 0,\ j,k \in \{1,2,3\},\ j \neq k. \tag{37}$$

If we know $c(\tilde{\mathbf{P}}_i \wedge e_\infty)$, we can find coordinates of $\tilde{\mathbf{P}}_i$ easily, because

$$\begin{aligned} \tilde{\mathbf{P}}_i \wedge e_\infty &= (p_{i1}e_1 + p_{i2}e_2 + p_{i3}e_3 + \frac{1}{2}\|\mathbf{p}_i\|^2 e_\infty + e_0) \wedge e_\infty \\ &= p_{i1}e_1 \wedge e_\infty + p_{i2}e_2 \wedge e_\infty + p_{i3}e_3 \wedge e_\infty + e_0 \wedge e_\infty. && (38)\end{aligned}$$

Remark 3.2 *The set* $\Pi_i = \{\pi_{i-1,i,L}, \pi_{i,i+1,L}, \pi_{i-1,i,R}, \pi_{i,i+1,R}\}$ *contains four planes, which intersect in one point. But generally four planes need not to have an intersection point in the space. In real applications it might happen that it the position of a camera is not exactly as we assume it, we get multiple possible position of point* $\tilde{\mathbf{P}}_i$*. The real position of point* $\tilde{\mathbf{P}}_i$ *has to be estimated by appropriate approximation.*

Remark 3.3 *Condition* $\pi_{ij} \wedge \pi_{ik} \neq 0$, $j,k \in \{1,2,3\}$, $j \neq k$ *grants various planes of three out of four planes from the set* Π_i. *Two planes from the set would be identical in instance if points* $\tilde{\mathbf{P}}_i, \tilde{\mathbf{P}}_{i+1}, \mathbf{O}_L$ *and* $\mathbf{O}_R$ *layed on one plane.*

Point $\tilde{\mathbf{P}}_8$ lays on planes $\pi_{7,8,L}, \pi_{7,8,R}$ and on lines

$$\mathbf{L}_{8,L} = I(\mathbf{O}_L \wedge \tilde{\mathbf{P}}_{8L} \wedge e_\infty), \tag{39}$$

$$\mathbf{L}_{8,R} = I(\mathbf{O}_R \wedge \tilde{\mathbf{P}}_{8R} \wedge e_\infty). \tag{40}$$

We will find a pair of points $\tilde{\mathbf{P}}_8 \wedge e_\infty$ as

$$c(\tilde{\mathbf{P}}_8 \wedge e_\infty) = I(\mathbf{L}_{8,L} \wedge \pi_{7,8,R}) \tag{41}$$

or

$$c(\tilde{\mathbf{P}}_8 \wedge e_\infty) = I(\mathbf{L}_{8,R} \wedge \pi_{7,8,L}). \tag{42}$$

So have found all points $\tilde{\mathbf{P}}_i, i \in \{0,1,2,\dots,8\}$, thus we know the final position of the machine.

4 Self calibration

In the previous chapter, we solved the problem of the binocular vision for a specific 6-DOF manipulator. Next we consider that we lost information about the exact position of the cameras for a short time due to some influences. Our goal is to calibrate the system using knowledge of the manipulator construction (proportions, movement possibilities), the current recording of the cameras and their approximate position. The manipulator construction and the current recording give all possible camera positions, figure 7 shows the possible focus positions. The solution is not unique, but if we know the position of the cameras at least approximately, we can choose a specific possible position of the cameras, which is closest to the approximate position with respect to the appropriate metric.

Note that the current recording is given by images $\mathbf{P}_{iL}, \mathbf{P}_{iR}, i \in \{0,1,\dots,8\}$ on projection planes. Consider coordinate system of left, resp. right camera in such a way, that its focus is placed to point $\mathbf{O} = fe_1 + \frac{1}{2}f^2 e_\infty + e_0$ and its images $\mathbf{P}_{iL}$,$\mathbf{P}_{iR}$ are projected to plane $\pi_{yz} = e_1$. In our procedure we are trying to find a rotor $R_{ROT} = \exp(-\frac{\alpha_{ROT}}{2}\mathbf{L}_{ROT})$, which represents rotation of coordinate systems to each other and translator T_{TRANS} representing their relative displacement. Resulting motor $M_v = T_{TRANS}R_{ROT}$, resp. $\tilde{M}_v = \tilde{R}_{ROT}\tilde{T}_{TRANS}$ will transfer a representation of geometric objects from one coordinate system to the other one. This is key for the following description of all possible camera positions.

4.1 Relative cameras rotation

At first we want to find rotor R_{ROT}, which describes the rotation of the right camera coordinate system to the left camera coordinate system. Note that points $\tilde{\mathbf{P}}_1, \tilde{\mathbf{P}}_2, \tilde{\mathbf{P}}_3, \tilde{\mathbf{P}}_4$ always form a rectangle in the space with respect to the possible machine position configurations. An intersection of lines $I(\mathbf{P}_{1L} \wedge \mathbf{P}_{3L} \wedge e_\infty)$ and $I(\mathbf{P}_{2L} \wedge \mathbf{P}_{4L} \wedge e_\infty)$, resp. $I(\mathbf{P}_{1R} \wedge \mathbf{P}_{3R} \wedge e_\infty)$ and $I(\mathbf{P}_{2R} \wedge \mathbf{P}_{4R} \wedge e_\infty)$ is an image of center point $\mathbf{S}$ of rectangle $\tilde{\mathbf{P}}_1, \tilde{\mathbf{P}}_2, \tilde{\mathbf{P}}_3, \tilde{\mathbf{P}}_4$ (see figure 3). Relative rotation of cameras can be found using diagonal vectors of the rectangle (normal

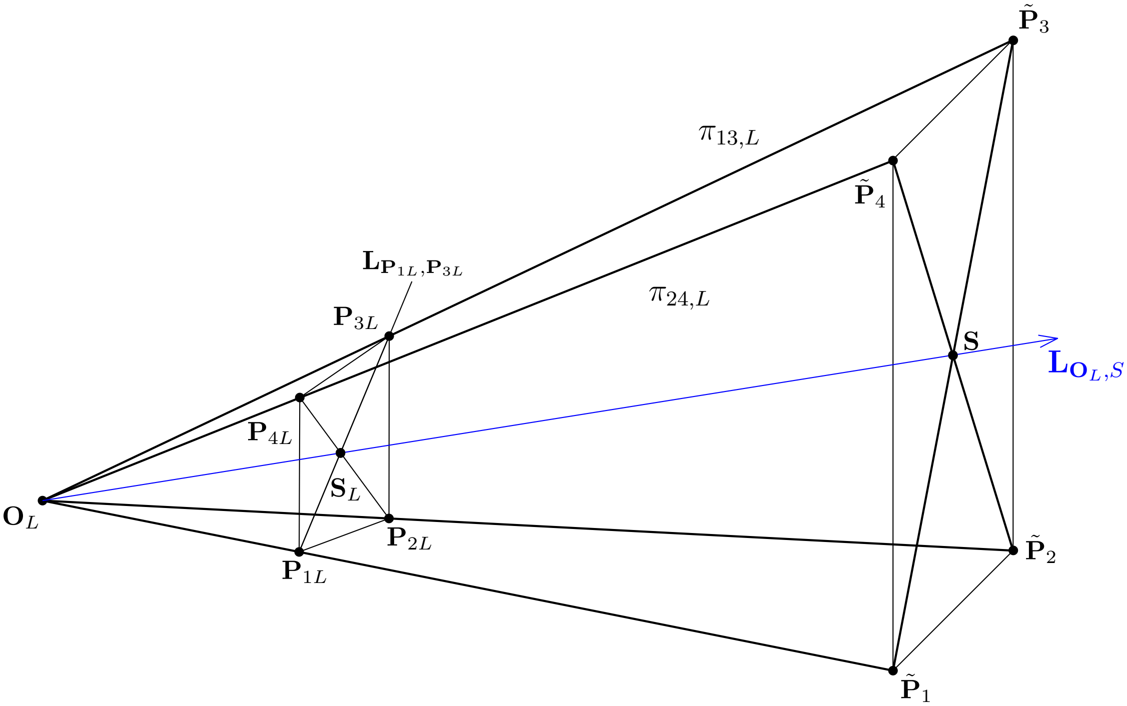


Figure 3: Calculation of line $\mathbf{L}_{\mathbf{O}_L,\mathbf{S}}$ $\mathbf{S}$ of rectangle $\tilde{\mathbf{P}}_1, \tilde{\mathbf{P}}_2, \tilde{\mathbf{P}}_3, \tilde{\mathbf{P}}_4$

vectors of symmetry planes $\tilde{\mathbf{P}}_3 - \tilde{\mathbf{P}}_1$ and $\tilde{\mathbf{P}}_4 - \tilde{\mathbf{P}}_2$) in the coordinate system for left and right cameras. Now consider coordinate system of the left camera. Firstly let us compute following objects

$$\begin{aligned} \pi_{13,L} &= \bullet(I(\mathbf{O} \wedge \mathbf{P}_{1L} \wedge \mathbf{P}_{3L} \wedge e_\infty)), && (43) \\ \pi_{24,L} &= \bullet(I(\mathbf{O} \wedge \mathbf{P}_{2L} \wedge \mathbf{P}_{4L} \wedge e_\infty)), && (44) \\ \mathbf{L}_{\mathbf{P}_{1L},\mathbf{P}_{3L}} &= \circ(I(\mathbf{P}_{1L} \wedge \mathbf{P}_{3L} \wedge e_\infty)). && (45) \end{aligned}$$

Image $\mathbf{S}_L$ of center point $\mathbf{S}$ is the intersection of plane $\pi_{24,L}$ and line $\mathbf{L}_{\mathbf{P}_{1L},\mathbf{P}_{3L}}$, which we can express in the equation

$$c(\mathbf{S}_L \wedge e_\infty) = I(\pi_{24,L} \wedge \mathbf{L}_{\mathbf{P}_{1L},\mathbf{P}_{3L}}). \tag{46}$$

Point $\mathbf{S}$ belongs to line

$$\mathbf{L}_{\mathbf{O}_L,\mathbf{S}} = \circ(I(\mathbf{O} \wedge \mathbf{S}_L \wedge e_\infty)) \tag{47}$$

For illustration see figure 3.

Remark 4.1 *The algorithm fails if images from points* $\tilde{\mathbf{P}}_1, \tilde{\mathbf{P}}_2, \tilde{\mathbf{P}}_3$ *and* $\tilde{\mathbf{P}}_4$ *of one camera belong to one line.*

Now we will compute representations of line through focus and points $\tilde{\mathbf{P}}_1, \tilde{\mathbf{P}}_2, \tilde{\mathbf{P}}_3, \tilde{\mathbf{P}}_4$:

$$\mathbf{L}_{\mathbf{O}_L,\mathbf{P}_i} = \circ(I(\mathbf{O} \wedge \mathbf{P}_{iL} \wedge e_\infty)), \quad i \in \{1,2,3,4\}. \tag{48}$$

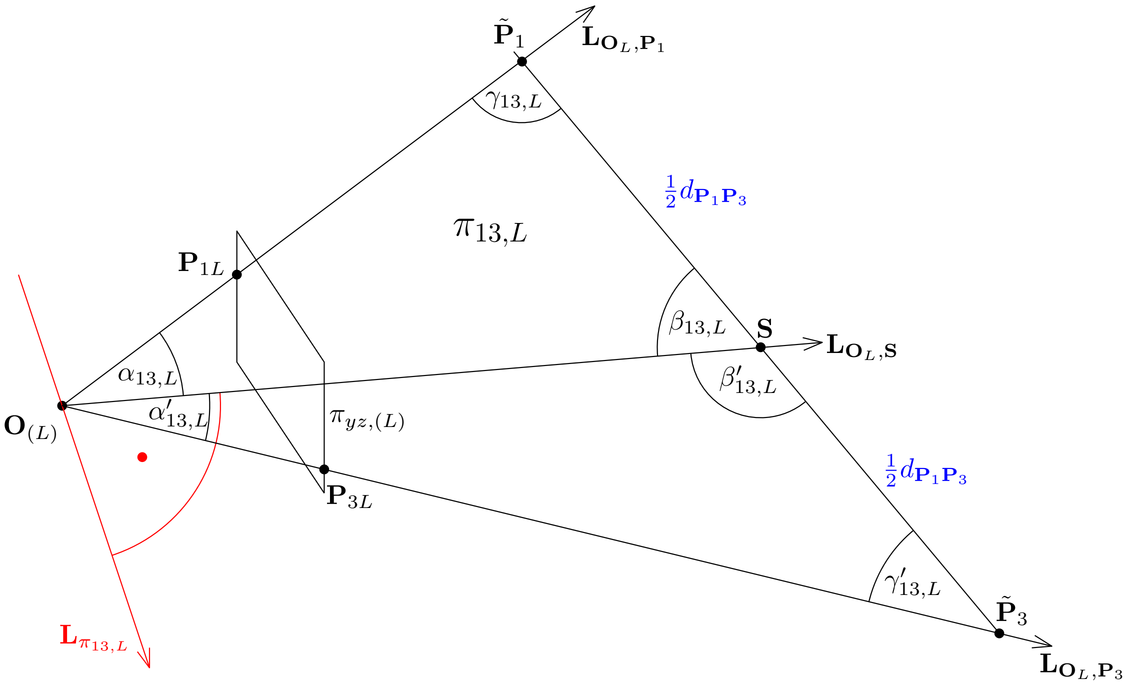

$\tilde{\mathbf{P}}_1$
$\mathbf{L}_{\mathbf{O}_L,\mathbf{P}_1}$
$\gamma_{13,L}$
$\frac{1}{2}d_{\mathbf{P}_1\mathbf{P}_3}$
$\pi_{13,L}$
$\mathbf{P}_{1L}$
$\beta_{13,L}$
$\mathbf{S}$
$\mathbf{L}_{\mathbf{O}_L,\mathbf{S}}$
$\alpha_{13,L}$
$\beta'_{13,L}$
$\alpha'_{13,L}$
$\pi_{yz,(L)}$
$\mathbf{O}_{(L)}$
$\mathbf{P}_{3L}$
$\frac{1}{2}d_{\mathbf{P}_1\mathbf{P}_3}$
$\gamma'_{13,L}$
$\tilde{\mathbf{P}}_3$
$\mathbf{L}_{\pi_{13,L}}$
$\mathbf{L}_{\mathbf{O}_L,\mathbf{P}_3}$

Figure 4: Illustration of angles given by lines in plane $\pi_{13,L}$

In the picture 4 we can see lines belonging to plane $\pi_{13,L}$, angles defining these lines and a line perpendicular to the plane. Analogously we can denote angles $\alpha_{24,L}, \alpha'_{24,L}, \beta_{24,L}, \beta'_{24,L}, \gamma_{24,L}, \gamma'_{24,L}$ in plane $\pi_{24,L}$. When computing directional vectors of rectangle $\tilde{\mathbf{P}}_1, \tilde{\mathbf{P}}_2, \tilde{\mathbf{P}}_3, \tilde{\mathbf{P}}_4$ diagonals we have to rotate line $\mathbf{L}_{\mathbf{O}_L,\mathbf{P}_3}$ with angle $\gamma'_{13,L}$ in plane $\pi_{13,L}$ and line $\mathbf{L}_{\mathbf{O}_L,\mathbf{P}_4}$ with angle $\gamma'_{24,L}$ in plane $\pi_{24,L}$. It is necessary to define angles $\gamma'_{13,L}, \gamma'_{24,L}$ and lines perpendicular to its respective planes. Firstly we will compute angles defined by line $\mathbf{L}_{\mathbf{O}_L,\mathbf{S}}$ and lines $\mathbf{L}_{\mathbf{O}_L,\mathbf{P}_i}, \in \{1,3\}$:

$$\alpha_{13,L} = \arccos(\mathbf{n}_{\mathbf{O}_L,\mathbf{S}} \cdot \mathbf{n}_{\mathbf{O}_L,\mathbf{P}_1}), \tag{49}$$

$$\alpha'_{13,L} = \arccos(\mathbf{n}_{\mathbf{O}_L,\mathbf{S}} \cdot \mathbf{n}_{\mathbf{O}_L,\mathbf{P}_3}), \tag{50}$$

where $\mathbf{n}_{\mathbf{O}_L,\mathbf{S}} = n_{S1}e_1 + n_{S2}e_2 + n_{S3}e_3$ is unitary directional vector of line $\mathbf{L}_{\mathbf{O}_L,\mathbf{S}}$ and $\mathbf{n}_{\mathbf{O}_L,\mathbf{P}_i} = n_{P_i1}e_1 + n_{P_i2}e_2 + n_{P_i3}e_3$ is unitary directional vector of line $\mathbf{L}_{\mathbf{O}_L,\mathbf{P}_i}$. According to law of sines we can infer that

$$\gamma'_{13,L} = \frac{\pi}{2} \quad \text{for} \quad \frac{\sin\alpha_{13,L}}{\sin\alpha'_{13,L}} + \cos(\pi - \alpha_{13,L} - \alpha'_{13,L}) = 0, \tag{51}$$

$$\gamma'_{13,L} = \arctan\frac{\sin(\pi - \alpha_{13,L} - \alpha'_{13,L})}{\frac{\sin\alpha_{13,L}}{\sin\alpha'_{13,L}} + \cos(\pi - \alpha_{13,L} - \alpha'_{13,L})} \quad \text{otherwise.} \tag{52}$$

Analogously we can compute angles $\alpha_{24,L}, \alpha'_{24,L}, \gamma_{24,L}, \gamma'_{24,L}$.

Remark 4.2 *Image of function* $\arctan(x)$ *in the previous expression we consider set* $\langle 0, \pi)/\{\frac{\pi}{2}\}$ *in order to get nonnegative, convex angles* $\gamma'_{13,L}, \gamma'_{24,L}$.

Now we will compute a line perpendicular to plane $\pi_{13,L}$, resp. $\pi_{24,L}$, passing the origin

$$\begin{aligned}
\mathbf{L}_{\pi_{13,L}} &= n_{1,\pi_{13,L}} e_2 \wedge e_3 + n_{2,\pi_{13,L}} e_3 \wedge e_1 + n_{3,\pi_{13,L}} e_1 \wedge e_2, & (53)\\
\mathbf{L}_{\pi_{24,L}} &= n_{1,\pi_{24,L}} e_2 \wedge e_3 + n_{2,\pi_{24,L}} e_3 \wedge e_1 + n_{3,\pi_{24,L}} e_1 \wedge e_2, & (54)
\end{aligned}$$

where $n_{i,\pi_{13,L}}$, resp. $n_{i,\pi_{24,L}}$ are coefficients of unitary normal vector of plane $\pi_{13,L}$, resp. $\pi_{24,L}$, $i \in \{1,2,3\}$. Next, let us perform following lines $\mathbf{L}_{\mathbf{O}_L,\mathbf{P}_3}$ a $\mathbf{L}_{\mathbf{O}_L,\mathbf{P}_4}$ rotations

$$\begin{aligned}
R_{13,L} &= \exp\left(-\frac{\gamma'_{13,L}}{2}\mathbf{L}_{\pi_{13,L}}\right), & (55)\\
R_{24,L} &= \exp\left(-\frac{\gamma'_{24,L}}{2}\mathbf{L}_{\pi_{24,L}}\right), & (56)
\end{aligned}$$

$$\begin{aligned}
\mathbf{L}_{P_{N13},L} &= R_{13,L}\mathbf{L}_{\mathbf{O}_L,\mathbf{P}_3}\tilde{R}_{13,L}, & (57)\\
\mathbf{L}_{P_{N24},L} &= R_{24,L}\mathbf{L}_{\mathbf{O}_L,\mathbf{P}_4}\tilde{R}_{24,L}. & (58)
\end{aligned}$$

Directional vectors of lines $\mathbf{L}_{P_{N13},L}$ and $\mathbf{L}_{P_{N24},L}$ refers to normal vectors of planes $\tilde{\mathbf{P}}_3 - \tilde{\mathbf{P}}_1$ and $\tilde{\mathbf{P}}_4 - \tilde{\mathbf{P}}_2$ in the coordinate system of left camera. In the direction of such vectors we can find points, whose distance from the origin is equal to 1. Those points we will denote as $\mathbf{P}_{N13,L}$ and $\mathbf{P}_{N24,L}$. Analogously we can find similar points $\mathbf{P}_{N13,R}$ and $\mathbf{P}_{N24,R}$ in the right coordinate system. Relative rotation of cameras coordinate systems describes a rotation rotating point $\mathbf{P}_{N13,R}$ into $\mathbf{P}_{N13,L}$ and at the same time $\mathbf{P}_{N24,R}$ into $\mathbf{P}_{N24,L}$ or conversely. If the rotation converts one point to another, then its rotational axis has to belong to a plane of symmetry of both points, because rotation is a circular motion, whose center is equidistant from both points, rotational axis passes the origin and is perpendicular to the plane, where the rotational circle belongs. Thanks to the expression (5) we can find a plane of symmetry of points $\mathbf{P}_{N13,L}$ and $\mathbf{P}_{N13,R}$ and a plane of symmetry of points $\mathbf{P}_{N24,L}$ and $\mathbf{P}_{N13,R}$ as

$$\begin{aligned}
\pi_{S13} &= \bullet(\mathbf{P}_{N13,L} - \mathbf{P}_{N13,R}), & (59)\\
\pi_{S24} &= \bullet(\mathbf{P}_{N24,L} - \mathbf{P}_{N24,R}) & (60)
\end{aligned}$$

Final rotational axis belongs to both of those planes therefore applies following

$$\overline{\mathbf{L}}_{ROT} = \circ(\pi_{S13} \wedge \pi_{S24}) \qquad (61)$$

Furthermore we need to compute rotational axes in such a way that final motor M_v converts representations of geometric objects from coordinate system of the right camera into coordinate system of the left camera. We can get an orientation of rotational axis in following way

$$\begin{aligned}
\pi_o &= I(e_0 \wedge \mathbf{P}_{N13,R} \wedge \mathbf{P}_{N13,L} \wedge e_\infty), & (62)\\
\mathbf{L}_{ROT} &= \operatorname{sgn}(\pi_o \cdot \overline{\mathbf{n}}_{ROT})\overline{\mathbf{L}}_{ROT}, & (63)
\end{aligned}$$

where $sgn(x)$ is sign function and $\overline{\mathbf{n}}_{ROT}$ is directional vector of line $\overline{\mathbf{L}}_{ROT}$. The meaning of the sign in the scalar product of vectors π_o and $\overline{\mathbf{n}}_{ROT}$ is visible in the picture 5.

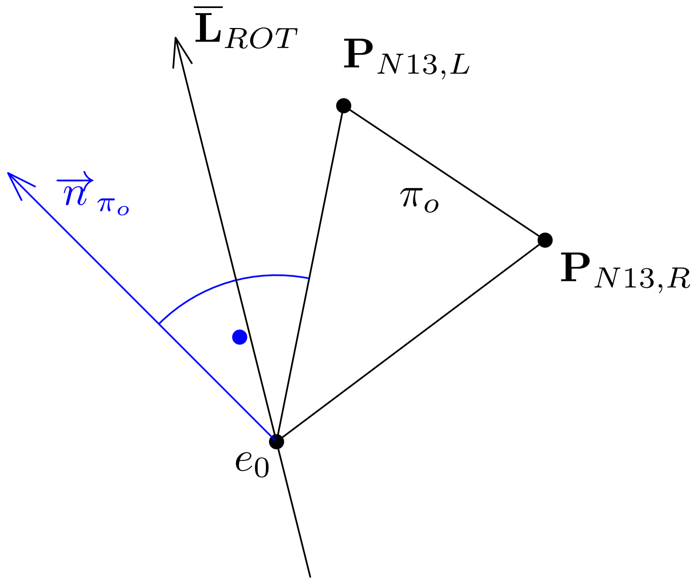


Figure 5: Normal vector of plane π_o defines orientation of rotational axis

Remark 4.3 *If $I(\mathbf{O} \wedge \mathbf{P}_{N13,R} \wedge \mathbf{P}_{N13,L} \wedge e_\infty) \cdot \overline{\mathbf{n}}_{ROT} = 0$, then it is a rotation of π radians, the axis orientation does not apply and we can use either $\mathbf{L}_{ROT} = \overline{\mathbf{L}}_{ROT}$ or $\mathbf{L}_{ROT} = -\overline{\mathbf{L}}_{ROT}$.*

Now we can compute the angle of rotation α_{ROT}. Rotation of point $\mathbf{P}_{N13,R}$ into $\mathbf{P}_{N13,L}$ is performed in a plane perpendicular to line $\overline{\mathbf{L}}_{ROT}$, where those points belong (see figure 6). Let us denote such a plane as π_p. To find plane π_p we have to compute auxiliary point $\mathbf{P}_{Np}$, which belongs to it in the following way

$$\mathbf{T}_p = 1 - \frac{1}{2}\pi_{24} e_\infty, \tag{64}$$

$$\mathbf{P}_{Np} = \mathbf{T}_p \mathbf{P}_{N13,L} \tilde{\mathbf{T}}_p. \tag{65}$$

Then we can compute plane π_p as

$$\pi_p = I(\mathbf{P}_{N13,L} \wedge \mathbf{P}_{N13,R} \wedge \mathbf{P}_{Np} \wedge e_\infty). \tag{66}$$

From the equation

$$c(\mathbf{P}_{ROT} \wedge e_\infty) = I(\pi_p \wedge \overline{\mathbf{L}}_{ROT}), \quad c \in \mathbb{R}/\{0\}, \tag{67}$$

we can compute an intersection $\mathbf{P}_{ROT}$ of plane π_p and line $\overline{\mathbf{L}}_{ROT}$ (see (38)). The angle of rotation which is defined by normal vectors of planes

$$\begin{aligned} \pi_{ROT,L} &= \mathbf{P}_{N13,L} - \mathbf{P}_{ROT}, & (68)\\ \pi_{ROT,R} &= \mathbf{P}_{N13,R} - \mathbf{P}_{ROT}, & (69) \end{aligned}$$

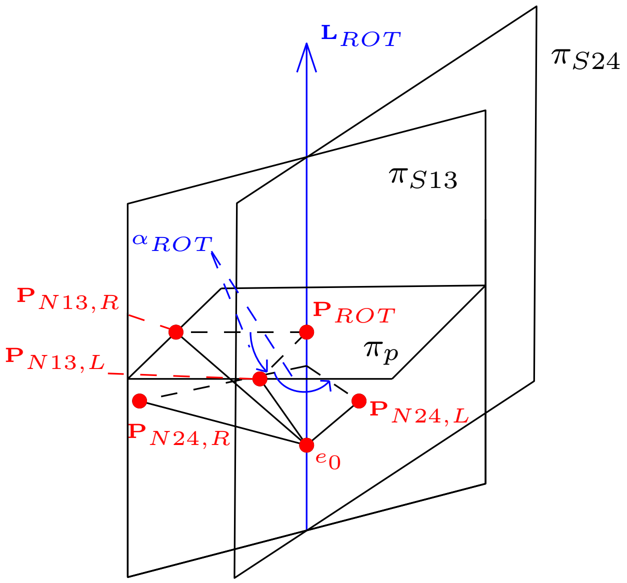


Figure 6: Rotation of point $\mathbf{P}_{N13,R}$ to point $\mathbf{P}_{N13,L}$ and another rotation of $\mathbf{P}_{N24,R}$ to $\mathbf{P}_{N24,L}$ around axis $\mathbf{L}_{ROT}$ with angle α_{ROT}. π_{S13} is a plane of symmetry of points $\mathbf{P}_{N13,R}$ and $\mathbf{P}_{N13,L}$, π_{S24} is a plane of symmetry of points $\mathbf{P}_{N24,R}$ and $\mathbf{P}_{N24,L}$

so

$$\alpha_{ROT} = \arccos(\pi_{ROT,L} \cdot \pi_{ROT,R}). \tag{70}$$

Next we will move rotation axis into focus $\mathbf{O}$

$$\begin{aligned} T_f &= 1 - \frac{1}{2} f e_1 e_\infty, & (71) \\ \tilde{\mathbf{L}}_{ROT} &= T_f \mathbf{L}_{ROT} \tilde{T}_f. & (72) \end{aligned}$$

We got the resulting rotor R_{ROT}, describing a rotation of the right camera coordinate system with respect to the left camera coordinate system as

$$R_{ROT} = \exp(-\frac{\alpha_{ROT}}{2} \tilde{\mathbf{L}}_{ROT}). \tag{73}$$

4.2 Relative shift of cameras

As the next step we have to find translator T_{TRANS}, which describes the shift of the right camera coordinate system to the left camera coordinate system. At first we have to get the length of rectangle $\tilde{\mathbf{P}}_1, \tilde{\mathbf{P}}_2, \tilde{\mathbf{P}}_3, \tilde{\mathbf{P}}_4$ diagonal as

$$d_{\mathbf{P}_1\mathbf{P}_3} = \sqrt{-2(\mathbf{P}_1 \cdot \mathbf{P}_3)}. \tag{74}$$

Next we want to compute a position of point $\tilde{\mathbf{P}}_3$ in the left camera coordinate system and the right camera coordinate system simultaneously. It holds that

$$\beta'_{13,L} = \pi - \alpha'_{13,L} - \gamma'_{13,L} \tag{75}$$

(see figure 4). From the law of sines it follows that the distance of point $\tilde{\mathbf{P}}_3$ from the left focus we can compute as

$$d_{\mathbf{O}_L\tilde{\mathbf{P}}_3} = \frac{1}{2} d_{\mathbf{P}_1\mathbf{P}_3} \frac{\sin \beta'_{13,L}}{\sin \alpha'_{13,L}}. \tag{76}$$

With translator $T_{3,L}$, which we can get as

$$\begin{aligned} \mathbf{t}_{3,L} &= d_{\mathbf{O}_L\tilde{\mathbf{P}}_3} \mathbf{n}_{\mathbf{O}_L,\mathbf{P}_3}, \quad (77) \\ T_{3,L} &= 1 - \frac{1}{2}\mathbf{t}_{3,L} e_\infty, \quad (78) \end{aligned}$$

where $\mathbf{n}_{\mathbf{O}_L,\mathbf{P}_3}$ is unitary directional vector of line $\mathbf{L}_{\mathbf{O}_L,\mathbf{P}_3}$, we can compute a position of point $\tilde{\mathbf{P}}_3$ in the left camera coordinate system as

$$\tilde{\mathbf{P}}_3^L = T_{3,L} \mathbf{O} \tilde{T}_{3,L}. \tag{79}$$

Analogously we can compute a position of the point in the right camera coordinate system, which we will denote as $\tilde{\mathbf{P}}_3^R$. Next we will rotate this point with rotor R_{ROT}

$$\tilde{\mathbf{P}}_{3,ROT}^R = R_{ROT} \tilde{\mathbf{P}}_3^R \tilde{R}_{ROT}. \tag{80}$$

The shift of the right camera coordinate system to the left camera coordinate system describes translator T_{TRANS}, which we will get as

$$\begin{aligned} \pi_{TRANS} &= \tilde{\mathbf{P}}_3^L - \tilde{\mathbf{P}}_{3,ROT}^R, \quad (81) \\ T_{TRANS} &= 1 - \frac{1}{2}\pi_{TRANS} e_\infty. \quad (82) \end{aligned}$$

The final motor

$$M_v = T_{TRANS} R_{ROT} \tag{83}$$

converts representations of geometric objects from the right camera coordinate system to the left camera coordinate system. The following equation

$$\tilde{\mathbf{P}}_i^L = M_v \tilde{\mathbf{P}}_i^R \tilde{M}_v, \quad i \in \{0, 1, ..., 8\}. \tag{84}$$

holds. Conjugate motor

$$\tilde{M}_v = \tilde{R}_{ROT} \tilde{T}_{TRANS} \tag{85}$$

converts representations of geometric objects from the left camera coordinate system to the right camera coordinate system, so

$$\tilde{\mathbf{P}}_i^R = \tilde{M}_v \tilde{\mathbf{P}}_i^L M_v, \quad i \in \{0, 1, ..., 8\}. \tag{86}$$

4.3 Cameras positions possibilities

Now we want to describe a space of all possible cameras positions in original coordinate system. With respect to possible machine motions we know points $\tilde{\mathbf{P}}_0 = \mathbf{P}_0$ a $\tilde{\mathbf{P}}_1 = \mathbf{P}_1$ - their position does not change with machine motion. With inferred relations from previous sections we can compute their position in right, resp. left camera coordinate system and their distance from focuses. At first we will convert representation of left focus and image in the left projection plane from left camera coordinate system to right camera coordinate system

$$\mathbf{O}_L^R \;=\; \tilde{M}_v \mathbf{O} M_v,\ \mathbf{P}_{iL}^R = \tilde{M}_v \mathbf{P}_{iL} M_v, \quad i \in \{0, 1, ..., 8\}. \tag{87}$$

Representations of points $\tilde{\mathbf{P}}_0$ and $\tilde{\mathbf{P}}_1$ in right camera coordinate system follows from

$$\begin{aligned} c_0(\tilde{\mathbf{P}}_0^R \wedge e_\infty) &= I((I(\mathbf{O}_L^R \wedge \mathbf{P}_{0L}^R \wedge \mathbf{P}_{1L}^R \wedge e_\infty)) \wedge (I(\mathbf{O} \wedge \mathbf{P}_{0R} \wedge e_\infty))), && (88)\\ c_1(\tilde{\mathbf{P}}_1^R \wedge e_\infty) &= I((I(\mathbf{O}_L^R \wedge \mathbf{P}_{0L}^R \wedge \mathbf{P}_{1L}^R \wedge e_\infty)) \wedge (I(\mathbf{O} \wedge \mathbf{P}_{1R} \wedge e_\infty))), && (89)\end{aligned}$$

where $c_0, c_1 \in \mathbb{R}/\{0\}$. The distance of points $\tilde{\mathbf{P}}_0$ and $\tilde{\mathbf{P}}_1$ from left and right camera focuses we get as

$$d_{\mathbf{O}_L \tilde{\mathbf{P}}_i} \;=\; \sqrt{-2(\mathbf{O}_L^R \cdot \tilde{\mathbf{P}}_i^R)},\ d_{\mathbf{O}_R \tilde{\mathbf{P}}_i} = \sqrt{-2(\mathbf{O} \cdot \tilde{\mathbf{P}}_i^R)}, \quad i \in \{0, 1\}. \tag{90}$$

Next fix the original coordinate system. Focus of left camera belongs to following two spheres

$$\mathbf{S}_{iL} = \tilde{\mathbf{P}}_i - \frac{1}{2} d^2_{\mathbf{O}_L \tilde{\mathbf{P}}_i} e_\infty, \quad i \in \{0, 1\}. \tag{91}$$

This yields that it belongs to circle

$$\mathbf{Z}_L = \mathbf{S}_{0L} \wedge \mathbf{S}_{1L}. \tag{92}$$

Analogously we can compute circle $\mathbf{Z}_R$, where belongs right camera focus. Now we will compute one particular position of left and right camera and we will label resulting points with upper index z. Let us combine plane $\pi_{xz} = e_2$ and circle $\mathbf{Z}_R$ and we get its intersection (pair of points) as

$$c(\mathbf{O}_R^z \wedge \mathbf{O}_R^{z_2}) = I(\mathbf{Z}_R \wedge \pi_{xz}), \quad c \in \mathbb{R}/\{0\}. \tag{93}$$

Now we want to choose point $\mathbf{O}_R^z$ out of pair of points $c(\mathbf{O}_R^z \wedge \mathbf{O}_R^{z_2})$ with positive x-axis coordinate. We can compute a line, which is defined by those points, as

$$\mathbf{L}_{\mathbf{O}_R^z \mathbf{O}_R^{z_2}} = I(c(\mathbf{O}_R^z \wedge \mathbf{O}_R^{z_2}) \wedge e_\infty). \tag{94}$$

Let $\mathbf{M}_{\mathbf{O}_R^z \mathbf{O}_R^{z_2}}$ be the midpoint of line segment $|\mathbf{O}_R^z \mathbf{O}_R^{z_2}|$ given by (6) and $d_{\mathbf{O}_R^z \mathbf{O}_R^{z_2}}$ is its length given by (8). We can get points $\mathbf{O}_R^z$ and $\mathbf{O}_R^{z_2}$ as a shift of point $\mathbf{M}_{\mathbf{O}_R^z \mathbf{O}_R^{z_2}}$ in positive, resp. negative orientation of line $\mathbf{L}_{\mathbf{O}_R^z \mathbf{O}_R^{z_2}}$ with displacement $\frac{1}{2} d_{\mathbf{O}_R^z \mathbf{O}_R^{z_2}}$. Furthermore we move points $\mathbf{P}_{0R}$ and $\mathbf{P}_{1R}$ in following way

$$\begin{aligned} \pi_{t_z} &= \mathbf{O}_R^z - \mathbf{O}, && (95)\\ T_z &= 1 - \frac{1}{2} \pi_{t_z} e_\infty, && (96)\\ \mathbf{P}_{iR,T} &= T_z \mathbf{P}_{iR} \tilde{T}_z, \quad i \in \{0, 1\}, . && (97)\end{aligned}$$

Now we need to determine a rotation of right camera with respect to the original position. To achieve that we will use analogical inference as in the picture 6. Initially we want to find images positions of points $\tilde{\mathbf{P}}^z_{0R}$ and $\tilde{\mathbf{P}}^z_{1R}$. These points belongs to lines

$$\mathbf{L}_{\mathbf{O}^z_R\tilde{\mathbf{P}}_i} \;=\; \circ(I(\mathbf{O}^z_R \wedge \tilde{\mathbf{P}}_i \wedge e_\infty)), \quad i \in \{0,1\}, \tag{98}$$

in distances

$$d_{\mathbf{OP}_{iR}} = \sqrt{-2(\mathbf{O}\cdot\mathbf{P}_{iR})}, \quad i \in \{0,1\}, \tag{99}$$

from focus $\mathbf{O}^z_R$. We can get images $\tilde{\mathbf{P}}^z_{0R}$ and $\tilde{\mathbf{P}}^z_{1R}$ with a shift of focus $\mathbf{O}^z_R$ with unitary directional vector of line $\mathbf{L}_{\mathbf{O}^z_R\tilde{\mathbf{P}}_0}$, resp. $\mathbf{L}_{\mathbf{O}^z_R\tilde{\mathbf{P}}_1}$ in the following way

$$\begin{aligned} \mathbf{t}_{\mathbf{O}^z_R\tilde{\mathbf{P}}_{iR}} &= d_{\mathbf{OP}_{iR}}\mathbf{n}_{\mathbf{O}^z_R\tilde{\mathbf{P}}_i}, && (100)\\ T_{\mathbf{O}^z_R\tilde{\mathbf{P}}_{iR}} &= 1 - \frac{1}{2}\mathbf{t}_{\mathbf{O}^z_R\tilde{\mathbf{P}}_{0R}}e_\infty, && (101) \end{aligned}$$

$$\tilde{\mathbf{P}}^z_{iR} \;=\; T_{\mathbf{O}^z_R\tilde{\mathbf{P}}_{iR}}\mathbf{O}^z_R\tilde{T}_{\mathbf{O}^z_R\tilde{\mathbf{P}}_{iR}}, \quad i \in \{0,1\}. \tag{102}$$

Now we need to determine rotation converting point $\mathbf{P}_{0R,T}$ to point $\tilde{\mathbf{P}}^z_{0R}$ and point $\mathbf{P}_{1R,T}$ to point $\tilde{\mathbf{P}}^z_{1R}$ as well. So the axis of rotation belongs to plane of symmetry of points $\tilde{\mathbf{P}}^z_{0R}$ and $\mathbf{P}_{0R,T}$ and simultaneously it belongs to plane of symmetry of points $\tilde{\mathbf{P}}^z_{1R}$ a $\mathbf{P}_{1R,T}$. We can compute those planes as

$$\pi^z_i \;=\; \tilde{\mathbf{P}}^z_{iR} - \mathbf{P}_{iR,T}, \quad i \in \{0,1\}. \tag{103}$$

Then we can get the rational axis within its orientation as

$$\begin{aligned} \overline{\mathbf{L}}^z_{ROT\mathbf{O}_R} &= \pi^z_0 \wedge \pi^z_1, && (104)\\ \mathbf{L}^z_{ROT\mathbf{O}_R} &= \circ(\mathrm{sgn}(I(\mathbf{O}^z_R \wedge \mathbf{P}_{0R,T} \wedge \tilde{\mathbf{P}}^z_{0R} \wedge e_\infty)\cdot\overline{\mathbf{n}}^z_{ROT\mathbf{O}_R})\overline{\mathbf{L}}^z_{ROT\mathbf{O}_R}), && (105) \end{aligned}$$

where $\overline{\mathbf{n}}^z_{ROT\mathbf{O}_R}$ is directional vector of line $\overline{\mathbf{L}}^z_{ROT\mathbf{O}_R}$. Next we want to know an angle of such rotation. Firstly we will find an intersection $\mathbf{P}^z_{ROT\mathbf{O}_R}$ of rotational axis and plane perpendicular to it, which contains points $\tilde{\mathbf{P}}^z_{0R}$ and $\mathbf{P}_{0R,T}$

$$\begin{aligned} T^z_p &= 1 - \frac{1}{2}\pi^z_1 e_\infty, && (106)\\ \mathbf{P}^z_p &= T^z_p\tilde{\mathbf{P}}^z_{0R}\tilde{T}^z_p, && (107) \end{aligned}$$

$$c(\mathbf{P}^z_{ROT\mathbf{O}_R} \wedge e_\infty) = I(I(\tilde{\mathbf{P}}^z_{0R} \wedge \mathbf{P}_{0R,T} \wedge \mathbf{P}^z_p \wedge e_\infty) \wedge \mathbf{L}^z_{ROT\mathbf{O}_R}). \tag{108}$$

Then the angle of rotation define normal vectors of planes

$$\begin{aligned} \pi^z_{ROT\mathbf{P}_{0R,T}} &= \mathbf{P}_{0R,T} - \mathbf{P}^z_{ROT\mathbf{O}_R}, && (109)\\ \pi^z_{ROT\mathbf{P}^z_{0R}} &= \tilde{\mathbf{P}}^z_{0R} - \mathbf{P}^z_{ROT\mathbf{O}_R}, && (110) \end{aligned}$$

so

$$\alpha_{ROT\mathbf{O}_R} \;=\; \arccos(\pi^z_{ROT\mathbf{P}_{0R,T}} \cdot \pi^z_{ROT\mathbf{P}^z_{0R}}). \tag{111}$$

Resulting rotation describes rotor

$$R^z_{ROT\mathbf{O}_R} \quad = \quad \exp(-\frac{\alpha_{ROT\mathbf{O}_R}}{2}\mathbf{L}^z_{ROT\mathbf{O}_R}). \tag{112}$$

For images in right camera holds

$$\tilde{\mathbf{P}}^z_{iR} = R^z_{ROT\mathbf{O}_R}\mathbf{P}_{iR,T}\tilde{R}^z_{ROT\mathbf{O}_R}, \quad i \in \{0, 1, ..., 8\}. \tag{113}$$

Representations of geometric objects from right camera coordinate system converts to the original coordinate system motor

$$M_z = R^z_{ROT\mathbf{O}_R}T_z. \tag{114}$$

Finally, because motor $\tilde{M}_v$ converts representations of geometric objects from left camera coordinate system to right camera coordinate system and following relations hold

$$\mathbf{O}^z_L \quad = \quad M_z\tilde{M}_v\mathbf{O}M_v\tilde{M}_z, \ \tilde{\mathbf{P}}^z_{iL} = M_z\tilde{M}_v\mathbf{P}_{iL}M_v\tilde{M}_z, \quad i \in \{0, 1, ..., 8\}. \tag{115}$$

Thus all possible camera positions can be described as

$$R_\theta = \exp(-\frac{\theta}{2}e_1 \wedge e_2), \tag{116}$$

$$\mathbf{O}^\theta_L \quad = \quad R_\theta\mathbf{O}^z_L\tilde{R}_\theta, \ \tilde{\mathbf{P}}^\theta_{iL} = R_\theta\tilde{\mathbf{P}}^z_{iL}\tilde{R}_\theta, \tag{117}$$

$$\mathbf{O}^\theta_R \quad = \quad R_\theta\mathbf{O}^z_R\tilde{R}_\theta, \ \tilde{\mathbf{P}}^\theta_{iR} = R_\theta\tilde{\mathbf{P}}^z_{iR}\tilde{R}_\theta, \quad i \in \{0, 1, ..., 8\}, \tag{118}$$

where parameter θ takes values of interval $\langle 0, 2\pi\rangle$ (see figure 7).

5 Conclusion

We considered a system given by a specific manipulator and two cameras that detect the manipulator position. Using CGA, We have proposed a method, which can compute all possible camera positions precisely with minimum initial knowledge - recording of the cameras and manipulator construction. It can be useful for self calibration, if we lose information about the exact position of the cameras for a short time due to some influences. The solution is not unique, but the selection of a concrete possible position will allow knowledge of the approximate position of the cameras (last known position). Although the algorithm is limited with a quality of images while image processing - structure identification, the method itself is quite effective, consisting with just a few equations, which enables online re-calibration of sensors. This means that the process can re-calibrate the sensors position prior further analysis on every image sample. Furthermore this method is applicable on wide range of structures, namely solid structures, which are supported by plane skeleton or plane frame. To reduce the dependence on given sensors images the research could continue adding perturbations and noise to the problem. The file containing the supporting code shall be available on `https://www.vut.cz/www_base/vutdisk.php?i=277353a744`.

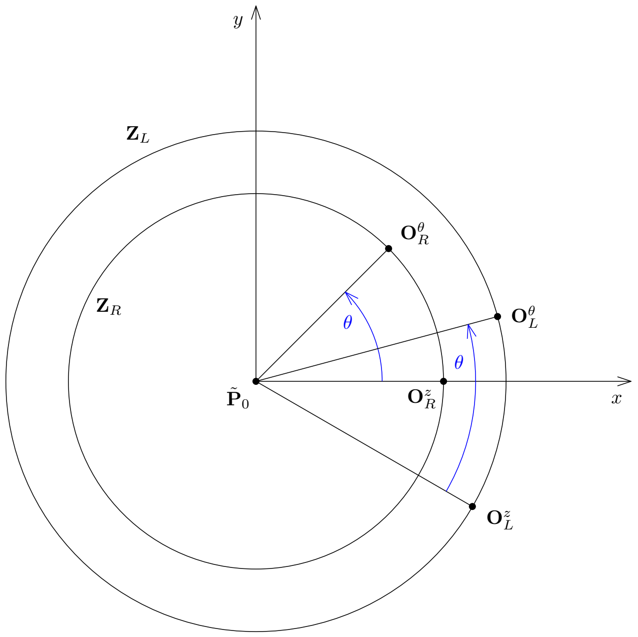


Figure 7: Final possible camera positions depending on parameter $\theta \in \langle 0, 2\pi \rangle$ - top view

References

[1] Bennett, S., Lasenby, J., Korkam, A., Iingiva, S. and Birkbeck, N.: Reconstruction of the pose of uncalibrated cameras via user-generated videos, *Proceedings of the 8th ACM/IEEE International Conference on Distributed Smart Cameras*, ICDSC 2014 (2014) (cited on p. 4)

[2] Dorst, L., Fontijne, D., Mann, S.: *Geometric Algebra for Computer Science: An Object-Oriented Approach to Geometry.* Morgan Kaufmann Publishers Inc. (2007) (cited on p. 2)

[3] Hadfield H., Wei L., Lasenby J.: The Forward and Inverse Kinematics of a Delta Robot. In: Magnenat-Thalmann N. et al. (eds) Advances in Computer Graphics. CGI 2020. Lecture Notes in Computer Science, vol 12221. Springer, Cham. (2020) (cited on p. 4)

[4] Hildenbrand, D.: *Foundations of Geometric Algebra Computing.* Springer Science & Business Media (2013) (cited on p. 2)

[5] Hildenbrand, D.: *Introduction to Geometric Algebra Computing.* CRC Press, Taylor & Francis Group (2019) (cited on p. 2)

[6] Hildenbrand, D., Hrdina, J., Návrat, A., Vašík, P. Local controllability of snake robots based on CRA, theory and practice. Advances in Applied Clifford Algebras, 30(1) (2020).doi:10.1007/s00006-019-1022-8 (cited on p. 4)

[7] Hrdina, J.; Vašík, P.: Notes on differential kinematics in conformal geometric algebra approach. In Advances in Intelligent Systems and Computing; Springer: Cham, Switzerland, Volume 378 (2015) 363–374. (cited on p. 2)

[8] Hrdina, J., Návrat, A.: Binocular computer vision based on conformal geometric algebra. Advances in Applied Clifford Algebras, 27(3), 1945-1959. (2017) doi:10.1007/s00006-017-0764-4 (cited on p. 4)

[9] Hrdina, J., Návrat, A., Vašík, P., Dorst, L.: Projective geometric algebra as a subalgebra of conformal geometric algebra. Advances in Applied Clifford Algebras, 31(2) (2021) doi:10.1007/s00006-021-01118-7 (cited on p. 2)

[10] González-Jiménez, L., Carbajal-Espinosa, O., Loukianov, A., Bayro-Corrochano, E. Robust pose control of robot manipulators using conformal geometric algebra. Advances in Applied Clifford Algebras, 24(2), 533-552. (2014) doi:10.1007/s00006-014-0448-2 (cited on p. 4)

[11] Kleppe, A. and Egeland, O., 2016. Inverse kinematics for industrial robots using conformal geometric algebra. Modeling, Identification and Control, 37(1), pp. 63-75. (cited on p. 4)

[12] Lounesto, P.: Clifford Algebra and Spinors. 2nd edn. CUP, Cambridge (2006). (cited on p. 2)

[13] Zamora-Esquivel, J., Bayro-Corrochano, E. Kinematics and diferential kinematics of binocular robot heads Proceedings - IEEE International Conference on Robotics and Automation, 2006, art. no. 1642337, pp. 4130-4135 (2006) (cited on p. 4)

[14] Jiang, G., Luo, M., Bai, K.: Optical positioning technology of an assisted puncture robot based on binocular vision. International Journal of Imaging Systems and Technology, 29(2), (2019) 180-190. doi:10.1002/ima.22303 (cited on p. 4)

[15] Perwass, Ch.: *Geometric Algebra with Applications in Engineering* (1st edn). Springer Verlag, 2009. (cited on p. 2)

[16] Stodola, M. 2019. Monocular Kinematics Based on Geometric Algebras. Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics). Vol. 11472 LNCS. (cited on p. 4)

[17] Tichy, R., 2020. Inverse Kinematics for the Industrial Robot IRB4400 Based on Conformal Geometric Algebra. In Modelling and Simulation for Autonomous Systems. MESAS 2019. Lecture Notes in Computer Science, vol 11995. Springer, Cham. (cited on p. 4)

[18] Xu, C., Wang, D., Huang, D., Yuan, P., Han, W. (2019). Self-location of unmanned aerial vehicle using conformal geometric algebra. Advances in Applied Clifford Algebras, 29(4) doi:10.1007/s00006-019-0992-x (cited on p. 4)

[19] Ganmin Zhu, Shimin Wei, Ying Zhang, Qizheng Liao, CGA-based novel modeling method for solving the forward displacement analysis of 3-RPR planar parallel mechanism, Mechanism and Machine Theory, Volume 168 (2022) 104595 (cited on p. 4)