Rational Trigonometry Applied to Robotics

Robot Kinematic Modeling using Rational Trigonometry

João Pequito Almeida

***Abstract*—This paper presents a short summary of the work done applying Rational Trigonometry to Robotics. Rational Trigonometry, as defined by N. J. Wildberger, provides a description of the separation of points (*Quadrance*) and lines (*Spread*) that is independent of reference frames.**

The work developed introduces a framework for using Rational Trigonometry concepts for the representation of points in a Reference Frame. It also introduces the Fixed Frames Model for Robot Kinematics, a model for Kinematics that can be applied to any robotic structure.

***Index Terms*—Rational Trigonometry, Robot Kinematics**

I. Introduction

The work developed applies some main concepts of Rational Trigonometry to generic kinematic modeling of robotic structures, namely, sets of interconnected joints with motors that have angular and radial motion.

The main motivation behind this study is that Rational Trigonometry allows for simpler expressions of common triangular problems without using circular functions. This fundamental difference allows for a different formulation of kinematic problems and can be seen as the key driving factor of this study.

Rational Trigonometry introduces two main concepts: *quadrance* and *spread*. Both relate to common trigonometrical concepts by the expressions of equations 1 and 2.

$$quadrance = distance^2 \tag{1}$$

$$spread = sine(line\ separation\ angle)^2 \tag{2}$$

Despite these expressions, all concepts of Rational Trigonometry have their own expressions independent of standard Trigonometry. For further development of these concepts please check the publications [Wildberger] and [Wildberger 2] or a short summary in [Almeida].

II. Approach

Rational Trigonometry concepts can no longer be added using standard sum. The approach was therefore to obtain an algebraic structure for quadrances and spreads that would allow for a reasoning similar to the one used with distances and angles, respectively. The operators obtained are provided in equations 3 and 4.

$$\begin{gathered}\textit{Let } A \textit{ and } B \textit{ be generic quadrances,} \\ A \boxplus B = (\sqrt{A} + \sqrt{B})^2 \\ A \boxminus B = (\sqrt{A} - \sqrt{B})^2\end{gathered} \tag{3}$$

$$\begin{gathered}\textit{Let } s_A \textit{ and } s_B \textit{ be generic spreads,} \\ s_A \obar s_B = (\sqrt{s_A(1-s_B)} + \sqrt{s_B(1-s_A)})^2 \\ s_A \ominus s_B = (\sqrt{s_A(1-s_B)} - \sqrt{s_B(1-s_A)})^2\end{gathered} \tag{4}$$

A key aspect of these objects is revealed when these operators are used, when a point in motion approaches the null coordinate zones of the Reference Frame (a quadrant edge in a 2D Cartesian Reference Frame for example), its motion is reflected back. This means that these objects are “trapped” in $\mathbb{R}_0^+$ due to the very nature of the operators developed and this fact can be (and is) exploited in the work developed.

III. Point description using Rational Trigonometry

In order to describe points, a framework was conceived that encompasses 3 solutions to 3 main problems. The first is how to represent trajectories when the operators used keep points “trapped” in a slice of space. The second is how to have these coordinates cover a full reference frame. The third is how to uniquely identify a point in a standard Reference Frame.

To solve the first, trajectories are represented as a Base Matrix that can be expanded to

any slice of space. In 3D for example, a base Matrix would represent the trajectory in one unsigned octant that could later be expanded to any octant by swapping per-coordinate signs).

To solve the second, a matrix that includes all sign changes in a reference frame was developed, called a Reference Frame Matrix, that relates the said unsigned Base Matrix to a signed Reference Frame. By multiplying the Reference Frame Matrix and the Base Matrix, all reflections of the trajectory slice are obtained.

To solve the third, since all trajectory slices are already available using the Reference Frame Matrices and Base Matrices, a simple Line Selecting Matrix is used.

The three concepts put together are the framework for point description developed in this work. Equations 5, 6 and 7 provide examples of a Base Matrix, a Reference Frame Matrix and a Line Selecting Matrix, respectively. Note that the polar substitution is used since the most common problems in Robot Kinematics involve rotational motion.

$$B = \begin{bmatrix} \sqrt{s_\theta R} & 0 \\ 0 & \sqrt{(1-s_\theta)R} \end{bmatrix} \tag{5}$$

$$\mathcal{R} = \begin{bmatrix} 1 & 1 \\ -1 & 1 \\ -1 & -1 \\ 1 & -1 \end{bmatrix} \tag{6}$$

$$L = \begin{bmatrix} 1 & 0 & 0 & 0 \end{bmatrix} \tag{7}$$

Using these three concepts, a point P is now fully represented by the expression provided in equation 8, hereon referred to as the Joint Equation.

$$P = L \times \mathcal{R} \times B \tag{8}$$

This expression will be applied to each joint of a Robot, hence the name. These matrices add flexibility and can be used to represent not only position but also attitude. Please check [Almeida] for a detailed description.

IV. Analyzing sets of points

Whenever the L matrix used has more than one row, multiple reflections of the trajectory are obtained simultaneously. This allows for greater flexibility in simulations but adds the problem of combining each reflection with every other whenever multiple joints are being simulated. To address this issue the set of expressions of equation 9 was developed. This set of expressions does the equivalent of a Cartesian Product of sets but each set is represented as a matrix and each tuple as a row of that matrix. Figure 1 depicts the indexing used and δ denotes the Kronecker Delta.

$$\begin{gathered} M_j = [\delta_{1j}\delta_{2j}\dots\delta_{rank(L_i),j}] \\ CC(a,b) = \prod_{i=a}^{b} rank(L_i) \\ I = CC(1,i-1);\ O = CC(i+1,n) \\ P_i^* = \left(\sum_{j=1}^{rank(L_i)} \left(M_j \times P_i \otimes \mathbb{1}_{I,1} \otimes M_j^T \right) \right) \otimes \mathbb{1}_{O,1} \end{gathered} \tag{9}$$

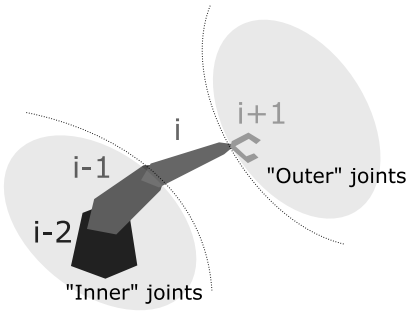

Figure 1. Depiction of the notation used for joint indexing

This process will be referred to as the $perm$ function. The resulting points $P_i^* = perm(P)$ have all the unique per-row combinations of the original points P_i. Even though this might seem an complex operation, it is rarely used as usually $rank(L_i) = 1$ and therefore $P_i = P_i^*$. A consequence of this expression is that this system for representing points can cover a full workspace.

V. The Fixed Frames Model for Robot Kinematics

Due to the particular nature of the point description used, transformation matrices turned

out to be too complex and a different approach was needed. From that necessity emerged the Fixed Frames Model and it is one of the most important consequences of the work developed.

The Fixed Frames Model (FFM) is, as the name indicates, a model that keeps Reference Frames fixed in terms of orientation (Reference Frames still move but they do not rotate). Traditionally, one would attach reference frames to key points of a robot and then transform them using transformation matrices. This model still attaches reference frames to key points of a robot but does not transform them during motion. This difference allows for a linear formulation of the kinematics problem, best exemplified by a robotic snake with n joints. The expressions for that example are provided in equation 10, where P_i has the Joint Equation values and P_{f_i} has the final coordinates referring to the workspace used and only one reflection per joint is considered. This formulation is only possible because Reference Frames do not move and the algebraic structure of the point description used is respected.

$$\begin{cases} P_{f_1} = P_1^* \\ P_{f_2} = P_1^* + P_2^* \\ \vdots \\ P_{f_n} = P_1^* + P_2^* + \ldots + P_n^* \end{cases}$$

$$\rightsquigarrow \begin{bmatrix} P_{f_1} \\ P_{f_2} \\ \vdots \\ P_{f_n} \end{bmatrix} = \begin{bmatrix} 1 & 0 & \ldots & 0 \\ 1 & 1 & \ldots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 1 & 1 & \ldots & 1 \end{bmatrix} \times \begin{bmatrix} P_1^* \\ P_2^* \\ \vdots \\ P_n^* \end{bmatrix} \quad (10)$$

The same type of analysis can be done for any robotic structure and all structures share an expression, the Structure Equation, provided in equation 11.

$$P_f = \mathcal{S} \times P^* \quad (11)$$

Note that even though the point description used is the one introduced in this work, any point description can be used as long as it places each coordinate in a column, each tuple in a row and all elements have a sum operator for use in the FFM.

VI. Representation of angular and sensor data

Since sensors move whenever the joints move, their values aren't referred to the fixed reference frame. To correct them so that they can be fed to the Joint Equation and Structure Equation, the expression of equation 12 can be used. Θ_c are the converted angles, Θ_s the sensor angles and Θ_{offset} the offset angles of the same sensors.

$$\Theta_c = \mathcal{C} \times \Theta_s + \Theta_{offset} \quad (12)$$

It is simply a correction matrix and an offset vector. This equation is applied directly to sensor data and therefore is called the Sensor-Actuator Equation. It is assumed, and that is the most common case, that the sensor data can be corrected using standard sum, if not, appropriate operators must be used. Typically a $\mathcal{C}$ matrix has a structure that is similar to the $\mathcal{S}$ matrix of the Structure Equation and always has full rank. This is due to the fact that there is a one to one relation between the angles from sensors and the angles referred to the Fixed Frames and this relation is linear thanks to the linearity of angles. When a joint moves, the sensor attached to its tip will still read the same angle, but the angle in the Fixed Frame is different: it is the sum of both the sensor input and the angle of the joint in motion. The $\mathcal{C}$ is the mathematical description of that and therefore translates between regular sensor inputs and the angles needed for the FFM.

VII. Analysis of constraints

A key aspect of Kinematics Modelling is how to represent and analyze constraints. In the work developed the following constraints are considered:

- Workspace constraints that limit the possible poses of the robot in a given workspace;
- Actuator constraints that limit the command and sensor values to a limited range;
- Structural constraints limit the poses that a robot can assume due to connections between parts of its structure, limiting its motion.

The first two are the simplest and involve a simple logic test of the values being simulated, the last presents the greatest challenge and is discussed in detail in [Almeida].

For the first type of constraints, let W be the set of valid workspace points and P_f the set of all robot points in the workspace reference frame, P is valid if and only if $\forall_{P \in P_f} P \in W$.

For the second type of constraints, in a robot with n joints, let A_i be the set of valid angles of joint i and θ_i the angle for joint i. θ_i is valid if and only if $\forall_{i \in [1,n]} \theta_i \in A_i$.

For the third type of constraints, distances must be calculated between points. These distances, like the Quadrance or Euclidean Distance, depend on coordinate-wise differences of points. To obtain them, a distance vector D must be calculated using a matrix that computes the referred coordinate-wise differences. This matrix will be referred to as Δ. This vector D is then compared with a set of constraint values K according to a combinatory rule σ, yielding the expressions of equation 13. The combinatory rule σ is a simple column vector that selects which coordinates should be included in the constraints. For example, consider a 2D point with attitude ($\begin{bmatrix} x & y & \theta \end{bmatrix}$). If the constraint is only in terms of position, σ will be $\begin{bmatrix} 1 & 1 & 0 \end{bmatrix}^T$.

The expression $A \bullet B$ denotes the Hadamard product between matrices A and B.

$$\begin{aligned} D &= \Delta \times P_f \\ D \bullet D \times \sigma &= K \bullet K \end{aligned} \tag{13}$$

The final set of expressions obtained is provided in 14.

$$\begin{cases} \forall_{P \in P_f} P \in W \\ \forall_{i \in [1,n]} \theta_i \in A_i \\ D \bullet D \times \sigma = K \bullet K \end{cases} \tag{14}$$

Again, these are generic expressions for any robotic structure and allow for a generalized analysis of the constraints involved in a given robotic structure simulation.

VIII. Forward Kinematics

Perhaps the best way to summarize this model is to provide a step-by-step algorithm that begins with the data from the sensor readings obtained and ends with the final simulation points.

The key expressions are given by equations 15, 16, 17 and 18 and encompass all of the model developed throughout this work.

- The Joint Equation and $perm$ function:

$$P_i^* = perm(L_i \times \mathcal{R}_i \times B_i) \tag{15}$$

- The Structure Equation:

$$P_f = \mathcal{S} \times P^* \tag{16}$$

- The Sensor-Actuator Equation:

$$\Theta_c = \mathcal{C} \times \Theta_s + \Theta_{offset} \tag{17}$$

- The Constraint Equations:

$$\begin{cases} \forall_{P \in P_f} P \in W \\ \forall_{i \in [1,n]} \theta_i \in A_i \\ D \bullet D \times \sigma = K \bullet K \end{cases} \tag{18}$$

Finally, the following steps summarize a typical usage of the model using the coordinates described in this work:

1) Set up the system used:
   a) Define the Base Matrices for each joint by inspecting the actuators of the robot;
   b) Define the Reference Frame Matrices for each joint by inspecting the workspace;
   c) Define all constraints for the Constraint Equations by inspecting the Robot and the workspace;
   d) Define the Structure Matrix and the Conversion Matrix of the Structure Equation and the Sensor-Actuator Equation, respectively;
2) Simulate:
   a) Process sensor data with the Sensor Equation obtaining corrected angles;
   b) Obtain the simulation of each joint by applying the Joint Equation to each

joint and, if desired, combine the reflections with the $perm$ operation;
c) Combine all joint simulations by applying the Structure Equation;
d) Use the data as desired and repeat from step 2a if needed.

IX. A STANDARD ROB3 MODEL AND METHOD COMPARISON.

Consider a standard ROB3 robot. The joints considered are depicted on figure 2. All joints have rotation motion and 3D Cartesian Reference Frames are used for position and Euler Angles are used for attitude values. The following notation is used:

- $A \oplus B$, direct sum of matrices A and B;
- $\left[\begin{array}{c|c} A & B \end{array}\right]$, horizontal concatenation of matrices A and B;
- $\left[\frac{A}{B}\right]$, vertical concatenation of matrices A and B;
- $S^{-1}(s_\alpha)$, conversion of a spread s_α to an angle α;
- I_n, $n \times n$ identity matrix;
- $\mathbb{1}_{n,m}$, $n \times m$ matrix of ones;
- $\mathbb{O}_{n,m}$, $n \times m$ matrix of zeros;
- $\mathbb{L}_n$, $n \times n$ lower diagonal matrix of ones (e.g. $\mathbb{L}_4 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 \\ 1 & 1 & 1 & 0 \\ 1 & 1 & 1 & 1 \end{bmatrix}$).

Equation 19 presents the Base Matrix used for all joints (B) and equation 20 presents the Reference Frame Matrix used for all joints ($\mathcal{R}$). A lookup function is used to map sensor angles to the $L \times \mathcal{R} \times B$ point description.

$$B_{cartesian} = \begin{bmatrix} \sqrt{s_\theta s_\varphi R} & 0 & 0 \\ 0 & \sqrt{(1-s_\theta)s_\varphi R} & 0 \\ 0 & 0 & \sqrt{(1-s_\varphi)R} \end{bmatrix}$$

$$B_{euler} = \begin{bmatrix} S^{-1}(s_\theta) & 0 & 0 \\ 1 & 0 & 0 \\ 0 & S^{-1}(s_\varphi) & 0 \\ 0 & 1 & 0 \\ 0 & 0 & S^{-1}(s_\varphi) \\ 0 & 0 & 1 \end{bmatrix}$$

$$B = B_{cartesian} \oplus B_{euler} \tag{19}$$

$$\mathcal{R}_{cartesian} = \begin{bmatrix} 1 & 1 & 1 \\ -1 & 1 & 1 \\ -1 & -1 & 1 \\ 1 & -1 & 1 \\ 1 & 1 & -1 \\ -1 & 1 & -1 \\ -1 & -1 & -1 \\ 1 & -1 & -1 \end{bmatrix}$$

$$\mathcal{R}_{euler} = \begin{bmatrix} 1 & 0 & 1 & 0 & 1 & 0 \\ -1 & \pi & -1 & 0 & 1 & 0 \\ 1 & -\pi & -1 & 0 & -1 & \pi \\ -1 & 0 & 1 & 0 & -1 & \pi \\ 1 & 0 & -1 & \pi & -1 & 0 \\ -1 & \pi & 1 & -\pi & -1 & 0 \\ 1 & -\pi & 1 & -\pi & 1 & -\pi \\ -1 & 0 & -1 & \pi & 1 & -\pi \end{bmatrix}$$

$$\mathcal{R} = \left[\begin{array}{c|c} \mathcal{R}_{cartesian} & \mathcal{R}_{euler} \end{array}\right] \tag{20}$$

Table I has the data for each joint and the fixed angle to be fed to the Sensor-Actuator Equation. In order to represent blocks that vary only in attitude a zero radius block is added.

Table I
ROB3 JOINT DATA

Joint	Radius (mm^2)	θ (rad)	ϕ (rad)
1	275^2	free	0
2	200^2	0	free
3	130^2	0	free
4	130^2	0	free
5	0	$\frac{\pi}{2}$	free

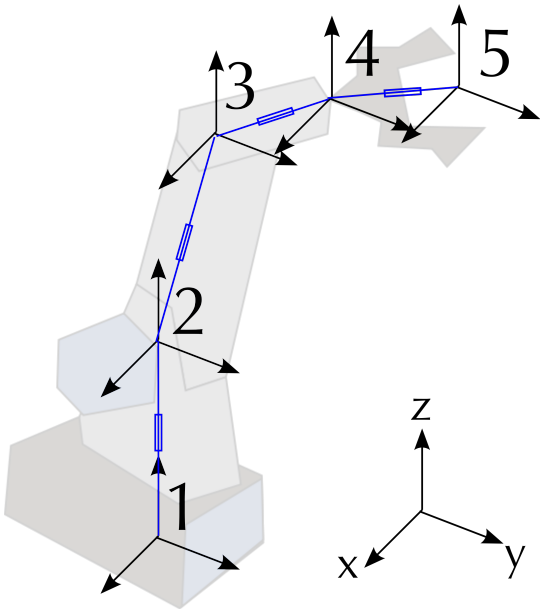

Figure 2. Joints considered for the ROB3 robot

Consider that the sensors read the angular data of equation 21. Fixed values are between parenthesis. These values are angles that depend on the joint structure and are not altered its motion.

$$\begin{bmatrix} \theta_1 & \varphi_1 \\ \theta_2 & \varphi_2 \\ \theta_3 & \varphi_3 \\ \theta_4 & \varphi_4 \\ \theta_5 & \varphi_5 \end{bmatrix} = \begin{bmatrix} \frac{\pi}{4} & (0) \\ (0) & \frac{\pi}{8} \\ (0) & -\frac{\pi}{4} \\ (0) & \frac{\pi}{3} \\ (\frac{\pi}{2}) & \pi \end{bmatrix} \tag{21}$$

Next, we apply the Sensor-Actuator Equation. First, for θ angles, only joint 5 is unaffected, so the correction needed is provided in equation 22.

$$\begin{bmatrix} \theta_{c_1} \\ \theta_{c_2} \\ \theta_{c_3} \\ \theta_{c_4} \end{bmatrix} = \mathbb{L}_4 \times \begin{bmatrix} \theta_1 \\ \theta_2 \\ \theta_3 \\ \theta_4 \end{bmatrix} = \begin{bmatrix} \frac{\pi}{4} \\ \frac{\pi}{4} \\ \frac{\pi}{4} \\ \frac{\pi}{4} \end{bmatrix} \tag{22}$$

Next, since the φ angles of joints 2, 3 and 4 are referred to the horizontal plane, so joint 1 has no effect on them. Therefore, the Conversion Matrix for φ affects only joints 2, 3 and 4. Therefore, only φ values need conversion, so the Sensor-Actuator Equation obtained is presented in equation 23.

$$\begin{bmatrix} \varphi_{c_2} \\ \varphi_{c_3} \\ \varphi_{c_4} \end{bmatrix} = \mathbb{L}_3 \times \begin{bmatrix} \varphi_2 \\ \varphi_3 \\ \varphi_4 \end{bmatrix} \tag{23}$$

Equation 23 yields the final corrected angular values of equation 24. These are the values to be converted to the new coordinate system. All values omitted remain identical ($\varphi_c = \varphi$).

$$\begin{bmatrix} \varphi_{c_2} \\ \varphi_{c_3} \\ \varphi_{c_4} \end{bmatrix} = \begin{bmatrix} \frac{\pi}{8} \\ -\frac{\pi}{8} \\ \frac{5\pi}{24} \end{bmatrix} \tag{24}$$

After the angular correction, the final angles are provided in equation 25.

$$\begin{bmatrix} \theta_1 & \varphi_1 \\ \theta_2 & \varphi_2 \\ \theta_3 & \varphi_3 \\ \theta_4 & \varphi_4 \\ \theta_5 & \varphi_5 \end{bmatrix} = \begin{bmatrix} \frac{\pi}{4} & 0 \\ \frac{\pi}{4} & \frac{\pi}{8} \\ \frac{\pi}{4} & -\frac{\pi}{8} \\ \frac{\pi}{4} & \frac{5\pi}{24} \\ (\frac{\pi}{2}) & \pi \end{bmatrix} \tag{25}$$

Now, applying the appropriate lookup function the L matrix and spreads are obtained. Usually the lookup function will be like the one provided in equation 26.

$$L_i = \begin{cases} \left[\begin{array}{cccccccc} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{array}\right] \\ if\ 0 \le \theta_i < \frac{\pi}{2}, -\frac{\pi}{2} \le \varphi_i < \frac{\pi}{2} \\ \left[\begin{array}{cccccccc} 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \end{array}\right] \\ if\ \frac{\pi}{2} \le \theta_i < \pi, -\frac{\pi}{2} \le \varphi_i < \frac{\pi}{2} \\ \left[\begin{array}{cccccccc} 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \end{array}\right] \\ if\ -\pi \le \theta_i < -\frac{\pi}{2}, -\frac{\pi}{2} \le \varphi_i < \frac{\pi}{2} \\ \left[\begin{array}{cccccccc} 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \end{array}\right] \\ if\ -\frac{\pi}{2} \le \theta_i < 0, -\frac{\pi}{2} \le \varphi_i < \frac{\pi}{2} \\ \left[\begin{array}{cccccccc} 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \end{array}\right] \\ if\ 0 \le \theta_i < \frac{\pi}{2}, \frac{\pi}{2} \le \varphi_i < \frac{3\pi}{2} \\ \left[\begin{array}{cccccccc} 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \end{array}\right] \\ if\ \frac{\pi}{2} \le \theta_i < \pi, \frac{\pi}{2} \le \varphi_i < \frac{3\pi}{2} \\ \left[\begin{array}{cccccccc} 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \end{array}\right] \\ if\ -\pi \le \theta_i < -\frac{\pi}{2}, \frac{\pi}{2} \le \varphi_i < \frac{3\pi}{2} \\ \left[\begin{array}{cccccccc} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{array}\right] \\ if\ -\frac{\pi}{2} \le \theta_i < 0, \frac{\pi}{2} \le \varphi_i < \frac{3\pi}{2} \end{cases}$$

$$s_{\theta_i} = \sin^2 \theta_i$$

$$s_{\varphi_i} = \sin^2 \varphi_i \tag{26}$$

So, using the referred lookup function, the table II has the values obtained.

Table II
L MATRICES AND s VALUES OBTAINED FROM THE LOOKUP FUNCTION.

Joint i	L_i	s_{θ_i}	s_{φ_i}
1	$\left[\begin{array}{cccccccc} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{array}\right]$	.5	0
2	$\left[\begin{array}{cccccccc} 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \end{array}\right]$	.5	0.14645
3	$\left[\begin{array}{cccccccc} 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \end{array}\right]$	.5	0.14645
4	$\left[\begin{array}{cccccccc} 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \end{array}\right]$	.5	0.37059
5	$\left[\begin{array}{cccccccc} 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \end{array}\right]$	1	0

Now, we can apply the Joint Equation to obtain the values of table III (transposed for better page fit).

Finally, apply the Structure Equation to the values obtained and the final actuator points are calculated, displayed in table IV.

Table III
JOINT EQUATION VALUES FOR THE ROB3 STRUCTURE.

Joint i	$P^T = (L_i \times R_i \times B_i)^T$
1	$\left[\begin{array}{c} \sqrt{.5 * 0} * 275 = 0\ (mm) \\ \sqrt{.5 * 0} * 275 = 0\ (mm) \\ \sqrt{1} * 275 = 275\ (mm) \\ \frac{\pi}{4}\ (rad) \\ 0\ (rad) \\ 0\ (rad) \end{array}\right]$
2	$\left[\begin{array}{c} \sqrt{.5 * 0.14645} * 200 = 54.120\ (mm) \\ \sqrt{.5 * (0.14645)} * 200 = 54.120\ (mm) \\ -\sqrt{(1 - 0.14645)} * 200 = -184.78\ (mm) \\ \frac{\pi}{4}\ (rad) \\ \frac{7\pi}{8}\ (rad) \\ 0\ (rad) \end{array}\right]$
3	$\left[\begin{array}{c} -\sqrt{.5 * 0.14645} * 130 = -35.178\ (mm) \\ -\sqrt{.5 * (0.14645)} * 130 = -35.178\ (mm) \\ -\sqrt{(1 - 0.14645)} * 130 = -120.10\ (mm) \\ \frac{\pi}{4}\ (rad) \\ -\frac{7\pi}{8}\ (rad) \\ 0\ (rad) \end{array}\right]$
4	$\left[\begin{array}{c} \sqrt{.5 * 0.37059} * 130 = 55.96\ (mm) \\ \sqrt{.5 * 0.37059} * 130 = 55.96\ (mm) \\ -\sqrt{(1 - 0.37059)} * 130 = -103.14\ (mm) \\ \frac{\pi}{4}\ (rad) \\ \frac{19\pi}{24}\ (rad) \\ 0\ (rad) \end{array}\right]$
5	$\left[\begin{array}{c} 0\ (mm) \\ 0\ (mm) \\ 0\ (mm) \\ 0\ (rad) \\ 0\ (rad) \\ \pi\ (rad) \end{array}\right]$

$$\left[\begin{array}{ccc} P_{f_x} & P_{f_y} & P_{f_z} \end{array}\right] = \mathbb{L}_5 \times \left[\begin{array}{ccc} P_x & P_y & P_z \end{array}\right]$$

$$\left[\begin{array}{c} P_{f_\alpha} \end{array}\right] = \left[\begin{array}{c|c} \mathbb{1}_{5,1} & \mathbb{O}_{5,4} \end{array}\right] \times \left[\begin{array}{c} P_\alpha \end{array}\right]$$

$$P_{f_\beta} = \left[\begin{array}{c} \left[\begin{array}{c|c} \mathbb{O}_{1,1} \oplus \mathbb{L}_3 & \mathbb{O}_{4,1} \end{array}\right] \\ \hline \left[\begin{array}{ccccc} 0 & 1 & 1 & 1 & 0 \end{array}\right] \end{array}\right] \times \left[\begin{array}{c} P_\beta \end{array}\right]$$

$$P_{f_\gamma} = \mathbb{O}_{4,4} \oplus \mathbb{1}_{1,1} \times \left[\begin{array}{c} P_\gamma \end{array}\right]$$

$$P_f = \left[\begin{array}{cccccc} P_{f_x} & P_{f_y} & P_{f_z} & P_{f_\alpha} & P_{f_\beta} & P_{f_\gamma} \end{array}\right] \tag{27}$$

As a comparison, here is the same data fed to a standard model using the Denavit-Hartenberg (D-H) method. Figure 3 has the reference frame definition and tables V and V have the D-H parameters and transformation matrices respectively. Angles θ_1 to θ_5 are the sensor angles. Distances L_1 to L_4 are the lengths of the rigid blocks ($L_1 = 275, L_2 = 200, L_3 = 130, L_4 = 130$).

Table IV
FINAL ACTUATOR POINTS OF THE ROB3 STRUCTURE.

Joint i	$P^T = (L_i \times R_i \times B_i)^T$
1	$\begin{bmatrix} 0\ (mm) \\ 0\ (mm) \\ 275\ (mm) \\ \frac{\pi}{4}\ (rad) \\ 0\ (rad) \\ 0\ (rad) \end{bmatrix}$
2	$\begin{bmatrix} 54.120\ (mm) \\ 54.120\ (mm) \\ 90.224\ (mm) \\ \frac{\pi}{4}\ (rad) \\ \frac{7\pi}{8}\ (rad) \\ 0\ (rad) \end{bmatrix}$
3	$\begin{bmatrix} 18.942\ (mm) \\ 18.942\ (mm) \\ -29.882\ (mm) \\ \frac{\pi}{4}\ (rad) \\ 0\ (rad) \\ 0\ (rad) \end{bmatrix}$
4	$\begin{bmatrix} 74.902\ (mm) \\ 74.902\ (mm) \\ -133.0162\ (mm) \\ \frac{\pi}{4}\ (rad) \\ \frac{19\pi}{24}\ (rad) \\ 0\ (rad) \end{bmatrix}$
5	$\begin{bmatrix} 74.902\ (mm) \\ 74.902\ (mm) \\ -133.0162\ (mm) \\ \frac{\pi}{4}\ (rad) \\ \frac{19\pi}{24}\ (rad) \\ \pi\ (rad) \end{bmatrix}$

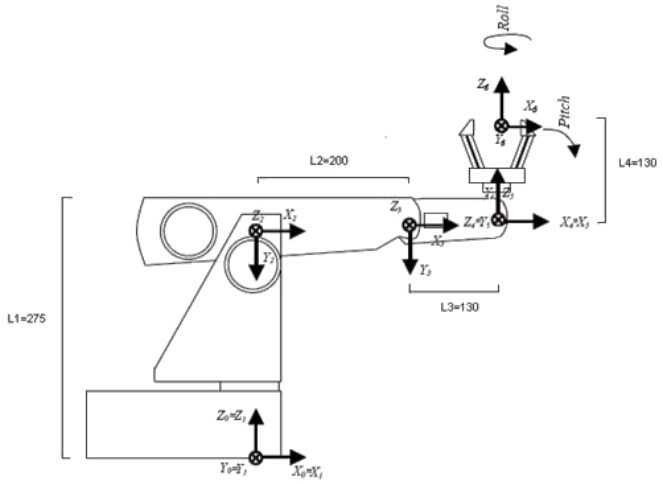

Figure 3. Reference Frame definition for ROB3 using D-H

Each point can now be obtained, the results are exactly the same as the ones using the FFM and are provided in table VII. The only difference is the indexing used for both. Attitude isn't included as only the final actuator was analyzed using this method.

Table V
D-H PARAMETERS FOR THE ROB3 STRUCTURE.

i	α_{i-1}	a_{i-1}	d_i	Θ_i
1	0	0	0	Θ_1
2	-90	0	0	Θ_2
3	0	L_2	0	Θ_3
4	0	L_3	0	Θ_4
5	90	0	0	Θ_5
6	0	0	L_4	0

Table VI
D-H INTERMEDIATE TRANSFORMATION MATRICES.

Matrix	Value
${}^1_0T$	$\begin{bmatrix} \cos\theta_1 & -\sin\theta_1 & 0 & 0 \\ \sin\theta_1 & \cos\theta_1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$
${}^1_0T$	$\begin{bmatrix} \cos\theta_1 & -\sin\theta_1 & 0 & 0 \\ \sin\theta_1 & \cos\theta_1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$
${}^2_1T$	$\begin{bmatrix} \cos\theta_2 & -\sin\theta_2 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ -\sin\theta_2 & -\cos\theta_2 & L_1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$
${}^3_2T$	$\begin{bmatrix} \cos\theta_1 & -\sin\theta_1 & 0 & 0 \\ \sin\theta_1 & \cos\theta_1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$
${}^4_3T$	$\begin{bmatrix} \cos\theta_3 & -\sin\theta_3 & 0 & L_2 \\ \sin\theta_3 & \cos\theta_3 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$
${}^5_4T$	$\begin{bmatrix} \cos\theta_4 & -\sin\theta_4 & 0 & L_3 \\ 0 & 0 & -1 & 0 \\ \sin\theta_4 & \cos\theta_4 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$
${}^6_5T$	$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & L_4 \\ 0 & 0 & 0 & 1 \end{bmatrix}$

Both of the methods were simulated and the performance of the D-H method was significantly better (average 150 times faster). This is due to the fact that the FFM has too much overhead when compared to a small set of transformation matrices. This tends to disappear as the robots grow larger and more complex and benchmarks done for snakes with many joints demonstrated that the FFM took about 50% the time to simulate than the standard counterpart as the number of joints increased.

Table VII
FINAL ACTUATOR POINTS OF THE ROB3 STRUCTURE USING THE D-H METHOD.

Joint i	$P^T = (L_i \times R_i \times B_i)^T$
1	$\begin{bmatrix} 0\ (mm) \\ 0\ (mm) \\ 0\ (mm) \end{bmatrix}$
2	$\begin{bmatrix} 0\ (mm) \\ 0\ (mm) \\ 275\ (mm) \end{bmatrix}$
3	$\begin{bmatrix} 54.120\ (mm) \\ 54.120\ (mm) \\ 90.224\ (mm) \end{bmatrix}$
4	$\begin{bmatrix} 18.942\ (mm) \\ 18.942\ (mm) \\ -29.882\ (mm) \end{bmatrix}$
5	$\begin{bmatrix} 18.942\ (mm) \\ 18.942\ (mm) \\ -29.882\ (mm) \end{bmatrix}$
6	$\begin{bmatrix} 74.902\ (mm) \\ 74.902\ (mm) \\ -133.0162\ (mm) \\ \frac{\pi}{4}\ (rad) \\ \frac{19\pi}{24}\ (rad) \\ \pi\ (rad) \end{bmatrix}$

X. CONCLUSIONS

The algorithm and model presented differs from common models as it attempts to provide a set of equations and algorithms that do not change from robot to robot. Whenever a new robot needs to be analyzed, all changes happen in the matrices ($\mathcal{S}$, $\mathcal{C}$, σ and Δ). This favors the analysis and development of new robotic structures, and due to the nature of the FFM, the complexity of a structure adds little to the complexity of the referred matrices. In fact, this model not only makes it easier to simulate complex structures but also favors them, both for Forward and Inverse Kinematics.

The Fixed Frames Model can also be applied using different coordinates, and though this is not explored, it adds to its flexibility.

This would add to the argument that Rational Trigonometry is only auxiliary in this work and not the central topic. Though this is true, the model would not have been possible without the initial change of representation and inherent difficulties.

Regarding Inverse Kinematics, due to the algebraic formulation obtained, it can be obtained by explicitly inverting the $\mathcal{S}$ and $\mathcal{C}$ matrices. If less than all the extremity points of the robot are desired, the inversion of the $\mathcal{S}$ matrix is an optimization problem by definition, and has optimal solutions described by its pseudo-inverse. This is currently under development and therefore no results are available.

The main gains obtained by using this model are mainly in terms of differences in paradigm. Complex structures are favored and the more flexible a robot is the better.

In terms of performance, the overhead is too high for simple structures but for complex structures like long snakes or complex parallel robots the model exhibits clear performance gains. For the case of a robotic snake benchmarks yielded a 50% decrease in simulation times. There are also a lot of possibilities for optimization, as the matrices have simple numeric values (usually 1, 0 and -1) and the whole algorithm hasn't been optimized at all.

All of these aspects are major gains in terms of both representation and implementation, the major loss being in terms of the size of the matrices and initial overhead.

A last, and also major gain, is the fact that this new representation gives a new and very different point of view of problems that have been studied for a long time, so hopefully this model can prove its value by the simple fact that it looks at old problems in a new way.

XI. FUTURE WORK AND PERSPECTIVES

A major work to be developed is the analysis of Inverse Kinematics. The problems that Inverse Kinematics presents are very interesting when put in terms of FFM and $L \times \mathcal{R} \times B$ coordinates and, as previously discussed, allow for globally optimal solutions for any structure.

Another work at hand is the development of the simulation toolkit that was used to make some of the calculations of this work. For more information please check [Almeida] or visit http://web.ist.utl.pt/ist152027/content/RTRTk/.

A final, and perhaps also key problem that isn't yet solved is to obtain the generic constraint solutions for parallel structures, if they exist.

REFERENCES

[Wildberger] Wildberger, N. J., "Divine Proportions: Rational Trigonometry to Universal Geometry", Wild Egg, 2005

[Wildberger 2] Wildberger, N. J., “A Rational Approach to Trigonometry”, http://web.maths.unsw.edu.au/ norman/papers/RationalTrig.pdf, 2005

[Almeida] Almeida, J. P., “Rational Trigonometry Applied to Robotics: Robot Kinematic Modelling using Rational Trigonometry”, http://web.ist.utl.pt/ist152027/content/tfc/, 2007