October 1988

A Computational Analysis of Line-Oriented Screw Transformations in Robotics

Janez Funda
University of Pennsylvania

University of Pennsylvania Department of Computer and Information Science Technical Report No. MS-CIS-88-83.

A Computational Analysis of Line-Oriented Screw Transformations in Robotics

Abstract

This paper contains a computational analysis and comparison of various representations of a general rigid body spatial screw displacement. Point transformations and line transformations are treated separately. In the context of point transformations, only a brief summary of the known techniques (*i.e.*, homogeneous transforms and quaternion/vector pairs) and their computational behavior is given. Among line transformations, which comprise the primary focus of this paper, four mathematical formalisms for effecting a general spatial screw displacement are presented and analyzed in terms of computational efficiency in performing (a) general screw displacements of lines, and (b) compositions of screw displacement operators. Both sequential and parallel algorithms are given for each operation. The four formalisms considered are: (1) dual orthogonal 3 x 3 matrix, (2) dual unit quaternion, (3) dual special unitary 2 x 2 matrix, and (4) dual Pauli spin matrices. The conclusion reached is that quaternion/vector pairs are the most economical of the point transformation operators, whereas dual unit quaternions represent the most compact and most efficient line transformation formalism.

Comments

University of Pennsylvania Department of Computer and Information Science Technical Report No. MS-CIS-88-83.

A COMPUTATIONAL ANALYSIS OF LINE-ORIENTED SCREW TRANSFORMATIONS IN ROBOTICS

Janez Funda

MS-CIS-88-83
GRASP LAB 159

Department of Computer and Information Science
School of Engineering and Applied Science
University of Pennsylvania
Philadelphia, PA 19104

October 1988

Acknowledgements: This material is based upon work supported by the National Science Foundation under Grants No. ECS-11879, IRI86-10617, MCS 8219196-CER, U.S. Army grants DAA29-84-K-0061, DAA29-84-9-0027

A Computational Analysis Of Line-Oriented Screw Transformations In Robotics

Janez Funda

Written Preliminary Examination
Part II

University of Pennsylvania
May 25, 1988

Abstract

This paper contains a computational analysis and comparison of various representations of a general rigid body spatial screw displacement. Point transformations and line transformations are treated separately. In the context of point transformations, only a brief summary of the known techniques (*i.e.*, homogeneous transforms and quaternion/vector pairs) and their computational behavior is given. Among line transformations, which comprise the primary focus of this paper, four mathematical formalisms for effecting a general spatial screw displacement are presented and analyzed in terms of computational efficiency in performing (a) general screw displacements of lines, and (b) compositions of screw displacement operators. Both sequential and parallel algorithms are given for each operation. The four formalisms considered are: (1) dual orthogonal 3×3 matrix, (2) dual unit quaternion, (3) dual special unitary 2×2 matrix, and (4) dual Pauli spin matrices. The conclusion reached is that quaternion/vector pairs are the most economical of the point transformation operators, whereas dual unit quaternions represent the most compact and most efficient line transformation formalism.

Contents

List of Figures

1 Introduction

The problem of finding mathematical tools to represent rigid body transformations in space has long been on the agenda of physicists and mathematicians, and is considered a well reasearched and well understood problem. However, with the advent of high-speed computers and their application to the generation of animating graphical images and control of robot manipulators, new interest arose in identifying compact and computationally efficient representations of spatial transformations.

The intent of this paper is to provide a survey of known line-based screw displacement formalisms and offer a computational analysis, contrasting and comparing the various representations in terms of their compactness (*i.e.*, storage requirements) and computational efficiency. This work is meant to complement the analysis of point transformations, contained in my masters thesis ([Funda,1988]), and thus provide a comprehensive view of the available mathematical models of spatial rigid body transformations.

The first part of the paper, Section 2, is dedicated to the review of the necessary mathematical preliminaries. The section contains a brief introduction to the concepts of dual numbers, Plücker line coordinates, dual angles, and screw displacements.

In Section 3 I present three methods of performing point transformations (Section 3.1) and four representations of a line-oriented general screw displacement (Section 3.2). Because a detailed analysis of point transformations is contained in a separate document ([Funda,1988]), only a brief introduction of the three formalisms is given and a summary of the computational trade-offs is reprinted for convenience from the reference. The four representations of line transformations are presented and discussed in much more detail. The development follows [Rooney,1978] with occasional exceptions, where I included my own formulations of the operators, usually for reasons of compactness and consistency.

Section 4 presents a rather detailed computational analysis and comparison of the four line transformation formalisms in terms of both sequential and parallel algorithms for two basic spatial operations: (1) a general spatial screw displacement of a line, and (2) composition of two general spatial screw displacement operators. Certain conventions regarding storage of the various operators are adopted and

explained. Section 4.4 contains a tabular summary of the sequential and parallel costs of the algorithms outlined in Sections 4.2 and 4.3, as well as a summary of the storage requirements for the four representations.

The last section (Section 5) surveys two approaches to solving the inverse kinematics problem for a general n degree of freedom (d.o.f.) robot manipulator, using line-oriented screw displacements. The first approach has been developed and demonstrated by Pennock and Yang ([Pennock,1985]). My discussion of the method closely follows the reference, with the exception of the fact that I chose to include some laboriously derived intermediate results, whose absence in the original paper makes the first reading a rather mystifying experience. The second approach to kinematic analysis of robot linkages represents my extension and adaptation of the method used by Yang and Freudenstein ([Yang,1964]) for analysis of closed-loop spatial mechanisms.

2 Mathematical Preliminaries

2.1 Dual Numbers

In an effort to motivate the mathematical origins of *dual numbers* as well as their interpretation in the context of planar and spatial geometry, I will introduce them as an algebraic system analogous to the familiar *ordinary complex numbers*. Rooney (in [Rooney,1978]) discusses both ordinary complex and dual numbers, along with a third type — *double numbers*, in the broader context of *generalized complex numbers* and attempts to interpret them both algebraically and geometrically. The interested reader is referred to [Rooney,1978] or [Yaglom,1968] for a rather interesting treatment of the topic.

The need to define complex numbers first arose in the context of investigating the roots of polynomial equations. The fact that even equation as simple as

$$x^2 + 1 = 0 \tag{1}$$

have no solution in the domain of real numbers lead mathematicians to invent the so called *imaginary unit* i, which satisfies Eq.(1) by definition, *i.e.*, $i^2 = -1$. A new algebraic structure, the *complex field* $\mathcal{C}$,

$$\mathcal{C} = \{a + ib : a, b \in \mathbb{R},\ i^2 = -1\} \tag{2}$$

was thus introduced as an extension of the reals and it was Gauss who in 1861 proved that this system was both necessary and sufficient to contain all roots of polynomial equations.

In analogy with the ordinary complex numbers, Clifford in 1873 introduced another type of complex numbers — *dual numbers* — of the form $x+\epsilon y$, where $x, y \in \mathbb{R}$ and ϵ is an algebraic unit with the property $\epsilon^2 = 0$ ([Clifford,1873]).[1] We refer to the components x and y of a dual number as the *real* and *dual* part, respectively. Addition of dual numbers is performed component-wise and a straightforward derivation (using the property $\epsilon^2 = 0$) reveals the following multiplication rule

$$(x + \epsilon y)(x' + \epsilon y') = xx' + \epsilon(yx' + xy') \tag{3}$$

The appropriate expressions for the *inverse*, *conjugate* and *norm* can be obtained in a straightforward manner. Furthermore, the *modulus* R and the *argument* T for a dual number $x + \epsilon y$ can be defined as follows (see Figure 1)

$$R = x \qquad ; \qquad T = \frac{y}{x} \text{ for } x \neq 0 \tag{4}$$

Consequently, using Taylor series expansion, we can define the *polar* and *exponential* representation of dual numbers as follows ([Rooney,1978]):

$$x + \epsilon y = R(1 + \epsilon T) = Re^{\epsilon T} \tag{5}$$

which is reminiscent of the corresponding expressions for the ordinary complex numbers, *i.e.*,

$$x + iy = r(\cos\theta + i\sin\theta) = re^{i\theta} \tag{6}$$

where $r = \sqrt{x^2 + y^2}$ and $\theta = \arctan\frac{y}{x}$.

Geometrically, both types of complex numbers can be treated as ordered pairs (x, y) and put into 1-1 correspondence with the cartesian planar points. Figure 1 shows this mapping and illustrates the planar geometric interpretation of the corresponding polar parameters r, θ, R and T.

Note that the parameter θ for the case of the ordinary complex number can be interpreted as either the angle between the x-axis and the radial line from the

[1] Note that the fact that $\epsilon^2 = 0$ does not imply that ϵ is a small or infintesimal quantity — rather, ϵ should be thought of as a nonzero imaginary unit with the stated property.

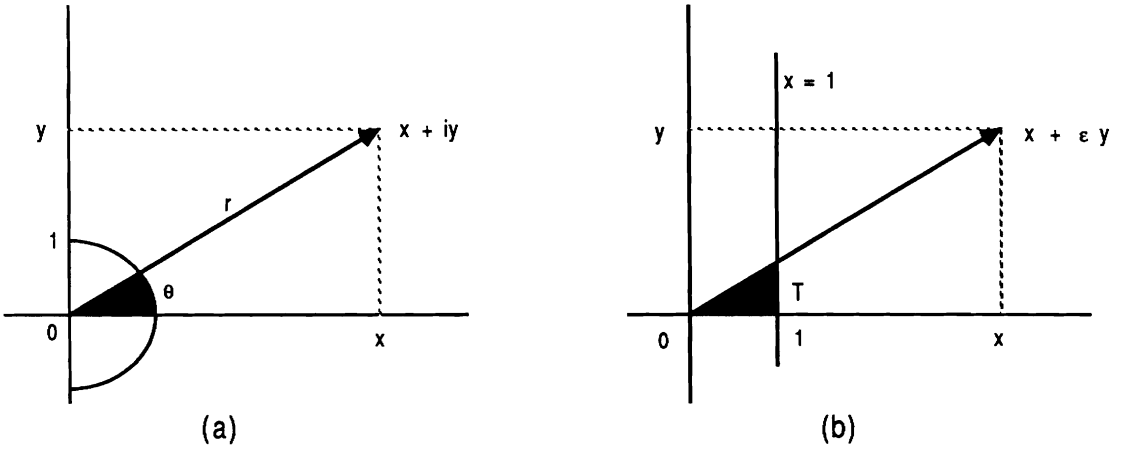

Figure 1: Interpretation of complex numbers as points in a plane: (a) ordinary complex number, and (b) dual number.

origin to the point $x + iy$, or as double the area of the shaded region in Figure 1a. Correspondingly, the parameter T of a dual number can be thought of as either the distance along the line $x = 1$ from the x-axis to the radial line from the origin to the point $x + \epsilon y$, or as double the area of the shaded region in Figure 1b.

While dual numbers don't seem to have an intuitive algebraic interpretation (unlike their ordinary complex number counterparts), they can be easily interpreted in a two-dimensional geometric sense as *planar operators*. It is well known that multiplication of a complex number $x + iy = re^{i\theta}$ (representing the planar point (x, y)) by the unit complex number $e^{i\alpha}$ *rotates* the point $x + iy$ counterclockwise about the origin through an angle α, *i.e.*,

$$e^{i\alpha}(re^{i\theta}) = re^{i(\alpha+\theta)} \tag{7}$$

The complex number $e^{i\alpha}$ thus serves as a planar rotational operator, rotating the points in the plane without affecting their distance from the origin.

Similarly, multiplying a dual number $x + \epsilon y = Re^{\epsilon T}$ by the unit dual number $e^{\epsilon\tau}$ gives

$$e^{\epsilon\tau}(Re^{\epsilon T}) = Re^{\epsilon(\tau+T)} \tag{8}$$

Referring to Figure 1b, we see that the unit dual number operator $e^{\epsilon\tau}$ effects a translation of the point (x, y) along the y-axis through a distance $R\tau$. The net

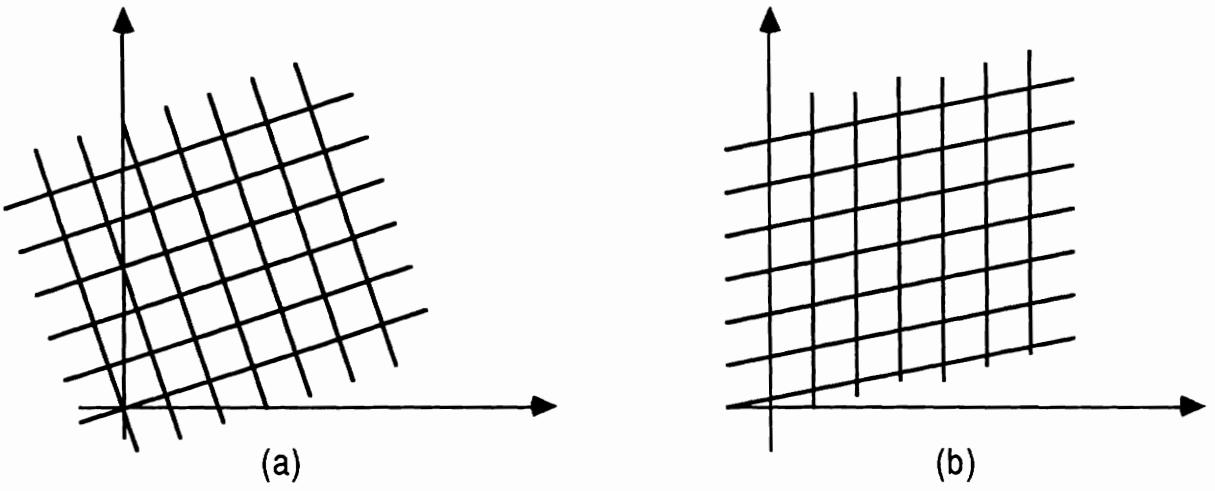

Figure 2: Plane transformations under multiplication by (a) ordinary complex number $e^{i\theta}$, and (b) dual number $e^{\epsilon\tau}$.

effect of applying this operator to the entire plane is therefore to *shear* the plane parallel to the y-axis through the *shear angle* of $\arctan\tau$. Figure 2 illustrates the effects of the unit complex number and unit dual number operators on a plane.

In conclusion to this introduction of dual numbers, let us briefly consider functions of dual arguments. If a given function f is at least once differentiable, then we can define its value for the dual argument $x + \epsilon y$ by expanding the corresponding Taylor series and setting $\epsilon^2 = \epsilon^3 = \cdots = 0$ ([Brand,1947]), *i.e.*,

$$f(x + \epsilon y) = f(x) + \epsilon y f'(x) \tag{9}$$

In particular,

$$\begin{aligned} \sin(x + \epsilon y) &= \sin x + \epsilon y \cos x \\ \cos(x + \epsilon y) &= \cos x - \epsilon y \sin x \end{aligned} \tag{10}$$

We will make much use of these results in later sections.

2.2 Plücker Coordinates

A point in space has 3 d.o.f. and consequently three-dimensional space is commonly treated as an aggregate of ∞^3 points. Alternatively, since we know that we need

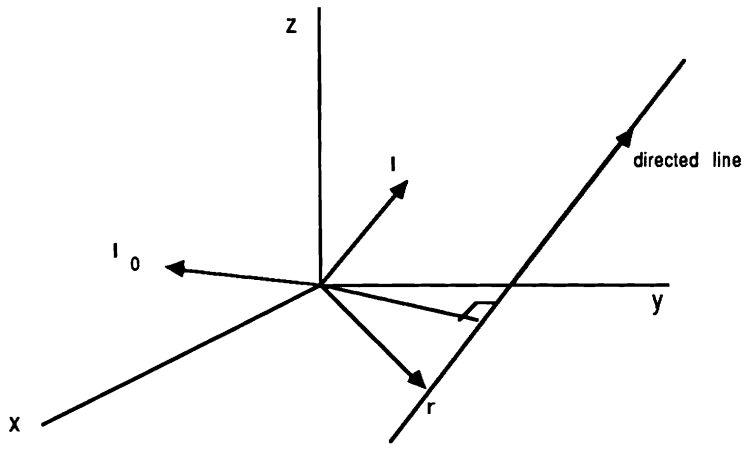

Figure 3: The Plücker coordinates of a line in space.

at least 4 parameters to completely specify a line in space (for instance, slope and intercept of the projection of the line onto two orthogonal planes), we could consider space to be composed of an aggregate of ∞^4 lines. Despite the extra d.o.f. we will find it useful and convenient to analyze spatial perturbances in terms of lines (rather than points), as lines arise naturally in the analysis of spatial mechanisms as rotational axes, link orientation axes, *etc.* We will be therefore in this paper concerned with *line transformations*, operating on (directed) lines in space.

Although only 4 parameters are needed to completely and uniquely define a line in space, we will for reasons of mathematical convenience use a system of coordinates where a line is described by 6 parameters, called *Plücker coordinates*. To ensure that the Plücker parameter space posesses only 4 d.o.f., we will impose two additional conditions onto the 6 parameters, thus effectively reducing the number of d.o.f. to 4, as desired.

A line is represented in Plücker coordinates in terms of two 3-D vectors, *i.e.*, (1) its *orientation* vector $\mathbf{l}$, and (2) its *moment* vector about the origin, $\mathbf{l_0} = \mathbf{r} \times \mathbf{l}$, where $\mathbf{r}$ is the position vector of an arbitrary point on the line (see Figure 3).

From the definition of the moment vector $\mathbf{l_0}$, we see that

$$\mathbf{l} \cdot \mathbf{l_0} = \mathbf{l} \cdot (\mathbf{r} \times \mathbf{l}) = 0 \tag{11}$$

i.e., the orientation vector $\mathbf{l}$ and the moment vector $\mathbf{l_0}$ are mutually orthogonal.

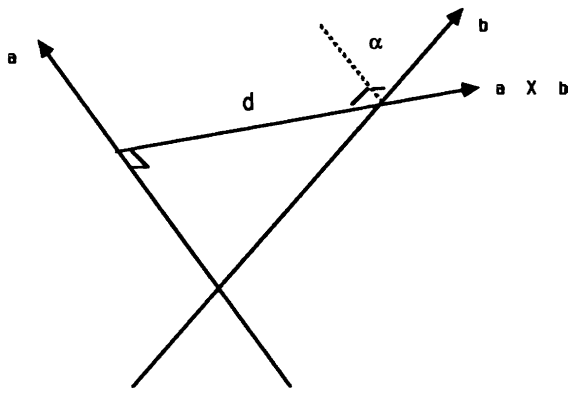

Figure 4: Definition of parameters describing relative location of two skew lines in space.

Also, since the vectors $\mathbf{l}$ and $k\mathbf{l}$ (for $k \neq 0$) define the same spatial orientation, we render the representation of a line unique by restricting the orientation vector to be of unit magnitude, *i.e.*,

$$\mathbf{l} \cdot \mathbf{l} = 1 \tag{12}$$

Note that this restriction makes the components of the orientation vector $\mathbf{l}$ correspond to the *direction cosines* of the line. The two conditions of Eqs.(11,12) together represent the necessary constraints onto the Plücker parameter space to restrict the number of d.o.f. of a line to 4.

Finally, note that for a line passing through the origin, the moment vector $\mathbf{l}_0$ is zero, and the line is completely determined by its orientation vector $\mathbf{l}$, as expected.

2.3 Dual Angles and Screws

In keeping with our ultimate aim of finding a line-oriented formalism for describing 3-D rigid body transformations, let us next look at the mathematics of describing the relative orientation of two skew lines in space, using the Plücker coordinates introduced above.

The relationship of two *directed* skew straight lines $\mathbf{a}$ and $\mathbf{b}$ in space can be expressed *uniquely* in terms of the following 3 parameters (Figure 4):

- the *common normal vector* $\mathbf{n} = \mathbf{a} \times \mathbf{b}$ between the two lines
- the *distance* d between the lines along the common normal, and
- the *twist angle* α, measured positive from $\mathbf{a}$ to $\mathbf{b}$

Note that d corresponds to the shortest perpendicular distance between the two lines. We now exploit the mathematical properties of *dual numbers* (Section 2.1, Eq.(10)) and the *Principle of Transference* (see Section 2.4) and combine the *twist angle* α and the *common normal distance* d into a *dual angle* $\hat{\alpha} = \alpha + \epsilon d$.[2] The advantages and theoretical justification of the dual angle notation were first investigated by the German mathematician Study ([Study,1901]).

The set of parameters defined above now allows us to specify the position and orientation of an arbitrary line l_2 in space with respect to a given line l_1 by a *single rotation* (through the twist angle α) about a *unique* spatial axis (the common normal $\mathbf{n}$), combined with a *unique translation* (through the distance d) along that same axis. This discovery provided the motivation for the famous theorem due to Chasles, which states that

> "any spatial displacement of a rigid body (*i.e.*, of the fixed coordinate frame associated with it) is equivalent to a combination of a *rotation* about and *translation* along some spatial axis"

This formulation of spatial transformations was later termed a *screw displacement* about a *screw axis* ([Ball,1900]) and this terminology has remained in use since.

In summary, a spatial screw displacement can be thought of as a *dual angular displacement* ($\hat{\alpha} = \alpha + \epsilon d$) about a spatial axis. The 6 Plücker coordinates of the screw axis along with the rotation angle α and the translation parameter d completely describe a general rigid body displacement. Finally, the ratio of the linear displacement to the rotation angle is sometimes referred to as the *pitch* of the screw ([Ball,1900],[Roth,1984]).

2.4 The Principle Of Transference

The most important result in the context of dual quantities is the famous *Principle of Transference*, due to [Krotkov,1895]. The principle relates *spherical* and *spatial*

[2]Note that for reasons of notational brevity the *hat symbol* ˆ is used to denote dual quantities.

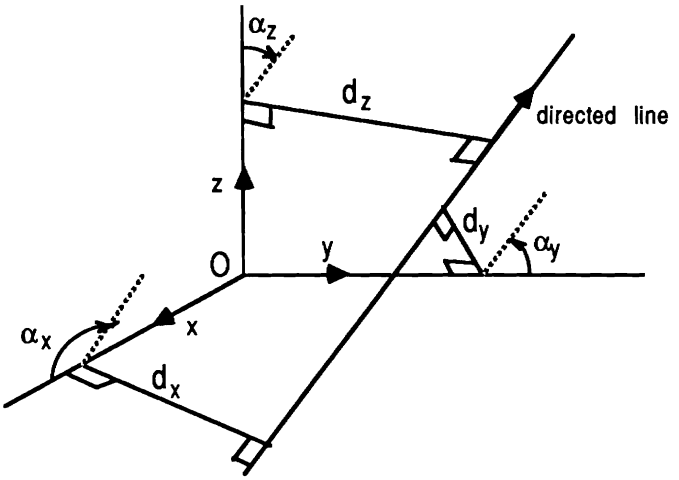

Figure 5: The dual direction cosines of a directed line in space.

geometry and states that

> "all laws and formulae relating to a *spherical configuration* (involving intersecting lines and real angles) are also valid when applied to an equivalent *spatial configuration* of skew lines, if each real angle α in the original formulae is replaced by the appropriate dual angle $\hat{\alpha} = \alpha + \epsilon d$" ([Rooney,1978])

Hence, a host of well-known results pertaining to *spherical geometry* can be carried over into the domain of *spatial geometry* by consistently replacing real angles with dual angles and real direction cosines with dual direction cosines.

2.5 Unit Line Vectors

A line in space (which does not pass through the origin) can be defined in terms of the 3 dual angles $\alpha_x + \epsilon d_x$, $\alpha_y + \epsilon d_y$, and $\alpha_z + \epsilon d_z$, which it makes with the three axes of the defining coordinate system (see Figure 5). Relating these dual angles to Plücker coordinates $(\mathbf{l}, \mathbf{l_0})$ yields (Section 2.1, Eq.(10)

$$\begin{aligned} \cos(\alpha_x + \epsilon d_x) &= \cos\alpha_x - \epsilon d_x \sin\alpha_x \\ \cos(\alpha_y + \epsilon d_y) &= \cos\alpha_y - \epsilon d_y \sin\alpha_y \\ \cos(\alpha_z + \epsilon d_z) &= \cos\alpha_z - \epsilon d_z \sin\alpha_z \end{aligned} \tag{13}$$

Recall from Section 2.2 that the components of the orientation vector $\mathbf{l} = \langle l_x, l_y, l_z \rangle$ correspond to the direction cosines of the line. Therefore we have

$$l_x = \cos\alpha_x \qquad l_y = \cos\alpha_y \qquad l_z = \cos\alpha_z \tag{14}$$

Similarly, it can be shown ([Rooney,1975]) that the following relationships hold for the components of the moment vector $\mathbf{l_0} = \langle l_{0x}, l_{0y}, l_{0z} \rangle$

$$l_{0x} = -d_x \sin\alpha_x \qquad l_{0y} = -d_y \sin\alpha_y \qquad l_{0z} = -d_z \sin\alpha_z \tag{15}$$

and we can hence write

$$\begin{aligned} \cos(\alpha_x + \epsilon d_x) &= l_x + \epsilon l_{0x} \\ \cos(\alpha_y + \epsilon d_y) &= l_y + \epsilon l_{0y} \\ \cos(\alpha_z + \epsilon d_z) &= l_z + \epsilon l_{0z} \end{aligned} \tag{16}$$

This suggests that we can specify a line in space as a *unit dual vector* $\hat{\mathbf{l}}$, whose components are the *dual direction cosines* of the line expressed in terms of Plücker coordinates, *i.e.*,

$$\hat{\mathbf{l}} = \mathbf{l} + \epsilon\mathbf{l_0} = \langle l_x + \epsilon l_{0x}, l_y + \epsilon l_{0y}, l_z + \epsilon l_{0z} \rangle \tag{17}$$

A dual vector of the form of Eq.(17) is commonly referred to as a *unit line vector*. A brief (but nontrivial) investigation of the properties of dual vectors yields some interesting results — let $\hat{\mathbf{a}} = \mathbf{a} + \epsilon\mathbf{a_0}$ and $\hat{\mathbf{b}} = \mathbf{b} + \epsilon\mathbf{b_0}$ be unit line vectors specifying two directed lines in space and let $\hat{\alpha} = \alpha + \epsilon d$ be the dual angle between them. Then we have ([Brand,1947], [Yang,1964])

$$\begin{aligned} &\textit{dot product}: && \hat{\mathbf{a}} \cdot \hat{\mathbf{b}} &&= \cos\hat{\alpha} \\ &\textit{cross product}: && \hat{\mathbf{a}} \times \hat{\mathbf{b}} &&= \hat{\mathbf{e}}\sin\hat{\alpha} \\ &\textit{algebraic product}: && \hat{\mathbf{a}}\hat{\mathbf{b}} &&= -(\hat{\mathbf{a}} \cdot \hat{\mathbf{b}}) + (\hat{\mathbf{a}} \times \hat{\mathbf{b}}) \\ & && &&= -\cos\hat{\alpha} + \hat{\mathbf{e}}\sin\hat{\alpha} \end{aligned} \tag{18}$$

where $\hat{\mathbf{e}}$ is the unit line vector along the common perpendicular of $\hat{\mathbf{a}}$ and $\hat{\mathbf{b}}$. Notice that these results are in complete analogy with the well-known corresponding spherical (*i.e.*, real vector) relationships obtained by "lifting the hats" in Eq.(18). These results therefore agree with and confirm the *Principle of Transference* of Section 2.4.

3 Spatial Transformations

A general spatial displacement of a rigid body consists of a finite translation along some spatial vector and a finite rotation about some axis. The rotational and translational axis in general need not bear any particular relationship to each other, and it is in fact often easiest and most intuitive to describe a spatial displacement as a combination of a rotation and translation where the two axes are not related. However, as we saw in Section 2.3, the combined effect of the two partial transformations (*i.e.*, rotation, translation) about their respective axes can be expressed as an equivalent *unique screw displacement* where the rotational and translational axes in fact coincide. Screws therefore offer an elegant formalism for describing spatial displacements. We will next consider some standard mathematical representations of screw displacements (Section 3) and investigate their computational properties (Section 4).

3.1 Point Transformations

We mentioned in Section 2.2 that space can be viewed in mathematical terms as composed of either *points* or *lines*. Adopting the first view, we then analyze spatial displacements in terms of *point transformations*, which operate on individual 3-D points. Point transformations are widely used in computer graphics and have been applied extensively to kinematic analyses of spatial mechanisms. Mathematical representations of point transformations tend to be more compact and more intuitive than their line counterparts, and thus tend to be more popular and better understood.

A general 3-D spatial displacement of a point vector can be described using a 4×4 *complex matrix* ([Rooney,1978]). However, although theoretically interesting, this representation is rather cumbersome and hence not suitable for applications where computational efficiency is an issue.

By far the most popular representation of point transformations is the 4×4 *real matrix* (also termed *homogeneous transform*), based on the idea of homogeneous coordinates, introduced by [Maxwell,1951]. A homogeneous transform is an extension

of the 3 x 3 real rotational matrix and has the form ([Lee,1982]):

$$\mathbf{T} = \begin{bmatrix} \mathbf{R}_{3\times3} & | & \mathbf{p}_{3\times1} \\ - - - & & - - - \\ \mathbf{f}_{1\times3} & | & s_{1\times1} \end{bmatrix} \tag{19}$$

where $\mathbf{R}_{3\times3}$ is the 3 x 3 real rotational matrix, $\mathbf{p}_{3\times1}$ is the linear displacement vector, the submatrix $\mathbf{f}_{1\times3}$ represents perspective transformation (set to zero for kinematic applications), and the lower right corner element $s_{1\times1}$ defines the global scaling factor (normally 1).[3]

The matrix $\mathbf{T}$ then operates on an arbitrary point vector $\mathbf{r}$ (expressed in homogeneous coordinates) by straightforward multiplication, *i.e.*,

$$\mathbf{r}' = \mathbf{T} * \mathbf{r} = (\mathbf{R}_{3\times3} * \mathbf{r}) + \mathbf{p} \tag{20}$$

where $\mathbf{r}'$ denotes the displaced vector $\mathbf{r}$.

Lastly, a less familiar representation of point transformations based on *unit quaternions* is possible (see Section 3.2.2 for the definition of a quaternion). This approach exploits the fact that unit quaternions can serve as non-stretching rotational operators (see Section 3.2.2, [Hamilton,1869]). Unfortunately, it turns out that quaternions can not be extended by use of homogeneous coordinates to include translational displacements ([Rooney,1978]). However, by augmenting the unit quaternion rotational operator $\mathbf{q}$ with a translational vector $\mathbf{p}$ and combining the two into a *quaternion/vector pair* operator $(\mathbf{q}, \mathbf{p})$, we can express a general spatial point transformation as follows ([Funda,1988]):

$$\mathbf{r}' = (\mathbf{q} * \mathbf{r} * \mathbf{q}^{-1}) + \mathbf{p} \tag{21}$$

Although less popular, this representation is as efficient, more compact, and more elegant than homogeneous transforms.

A detailed comparison of the latter two point transformation representations (*i.e.*, homogeneous transforms and quaternion/vector pairs) is the topic of my master's thesis ([Funda,1988]) and will hence not be considered here. Instead, the tabular summary of the computational behavior of the two formalisms is reproduced below for convenience.

[3]The point transformations considered in this section are not necessarily screw displacements, *i.e.*, the rotational and translational axes do not, in general, coincide.

Homogeneous Transforms *vs* Quaternion/Vector Pairs											
Operation	Norm	Sequential Execution						Parallel Execution			
		HT			Q/V			HT		Q/V	
		*	+	$\sqrt{}$	*	+	$\sqrt{}$	CPUs	cycles	CPUs	cycles
Spatial Trans	–	15	12	0	15	15	0	9	4	9	6
Composition	–	33	24	0	31	27	0	24	4	22	6
Inverse	–	15	9	0	15	12	0	6	5	9	5
Spatial Trans	$\surd$	45	28	2	23	18	1	9	16	9	11
Composition	$\surd$	63	40	2	39	30	1	24	16	22	11
Inverse	$\surd$	45	25	2	23	15	1	6	17	9	10

The reader is cautioned against direct comparisons between the above table of *point transformation costs*, and the tables summarizing *line transformation costs* in Section 4.4. Nevertheless, I feel that the inclusion of the above point transformation cost summary provides useful information.

In the remainder of this paper I will restrict my attention to line oriented representations of a general spatial screw displacement.

3.2 Line Transformations

Considering the Euclidean space to be composed of an aggregate of lines, rather than points, leads to *line-oriented* mathematical formulations of spatial displacements (*i.e.*, screws). Our mathematical vocabulary will therefore consist of dual vectors and dual angles, instead of points and real angles. In this section we will briefly review four distinct representations of screw displacements: (1) *dual orthogonal* 3×3 *matrix*, (2) *dual unit quaternion*, (3) *dual special unitary* 2×2 *matrix*, and (4) *dual Pauli spin matrices*.[4]

[4][Rooney,1978] actually discusses a fifth representation of a line-oriented screw displacement — *dual special unitary* 3×3 *matrix*. However, from the standpoint of computational efficiency, this clearly can not represent an improvement over the dual special unitary 2×2 matrix, and will be therefore omitted from this survey.

3.2.1 Dual Orthogonal 3 x 3 Matrix

As suggested by the name, this representation is derived from the real rotational matrix via an application of the *Principle of Transference*, which maps the 9 component real matrix into a dual matrix of the form

$$\hat{\mathbf{R}} = \begin{bmatrix} \hat{n}_x & \hat{o}_x & \hat{a}_x \\ \hat{n}_y & \hat{o}_y & \hat{a}_y \\ \hat{n}_z & \hat{o}_z & \hat{a}_z \end{bmatrix} \tag{22}$$

This matrix operates on an ordered triple of dual line coordinates

$$\hat{\mathbf{l}} = \langle \hat{l}_x, \hat{l}_y, \hat{l}_z \rangle = \langle l_x + \epsilon l_{0x}, l_y + \epsilon l_{0y}, l_z + \epsilon l_{0z} \rangle \tag{23}$$

and effects a screw displacement of the line $\hat{\mathbf{l}}$ into $\hat{\mathbf{l}}'$ as follows

$$\hat{\mathbf{l}}' = \hat{\mathbf{R}} * \hat{\mathbf{l}} \tag{24}$$

A general spatial displacement of a rigid body can be described using 6 independent parameters, corresponding to the 6 d.o.f. of the body (3 rotational, 3 translational). However, the dual orthogonal matrix of Eq.(22) describes a general screw displacement using 18 parameters — consequently, 12 orthogonality conditions must be satisfied to ensure that only 6 of the parameters are independent. These conditions are dual equivalents of the corresponding conditions imposed on the real 3 x 3 matrix, and can be stated concisely as the following 6 dual equations

$$\begin{aligned} \hat{n}_x^2 + \hat{n}_y^2 + \hat{n}_z^2 &= 1 \\ \hat{o}_x^2 + \hat{o}_y^2 + \hat{o}_z^2 &= 1 \\ \hat{a}_x^2 + \hat{a}_y^2 + \hat{a}_z^2 &= 1 \end{aligned} \tag{25}$$

$$\begin{aligned} \hat{n}_x\hat{o}_x + \hat{n}_y\hat{o}_y + \hat{n}_z\hat{o}_z &= 0 \\ \hat{o}_x\hat{a}_x + \hat{o}_y\hat{a}_y + \hat{o}_z\hat{a}_z &= 0 \\ \hat{a}_x\hat{n}_x + \hat{a}_y\hat{n}_y + \hat{a}_z\hat{n}_z &= 0 \end{aligned} \tag{26}$$

where Eq.(25) encodes the requirement that the three column dual vectors of the matrix be *unit dual vectors*, and Eq.(26) makes explicit the restriction that the same dual vectors be *mutually orthogonal.*

The dual matrix representing a screw displacement along an arbitrary spatial axis $\hat{\mathbf{n}}$ can again be derived from the corresponding real rotational matrix ([Paul,1981]) by replacing real quantities with the corresponding dual quantities. A screw displacement through a dual angle $\hat{\theta} = \theta + \epsilon d$ along the unit line vector $\hat{\mathbf{n}} = \langle \hat{\mathrm{n}}_x, \hat{\mathrm{n}}_y, \hat{\mathrm{n}}_z \rangle$ is then given by

$$\hat{\mathbf{R}}(\hat{\mathbf{n}}, \hat{\theta}) = \begin{bmatrix} \hat{\mathrm{n}}_x\hat{\mathrm{n}}_x \mathrm{ver}\,\hat{\theta} + \cos\hat{\theta} & \hat{\mathrm{n}}_y\hat{\mathrm{n}}_x \mathrm{ver}\,\hat{\theta} - \hat{\mathrm{n}}_z \sin\hat{\theta} & \hat{\mathrm{n}}_z\hat{\mathrm{n}}_x \mathrm{ver}\,\hat{\theta} + \hat{\mathrm{n}}_y \sin\hat{\theta} \\ \hat{\mathrm{n}}_x\hat{\mathrm{n}}_y \mathrm{ver}\,\hat{\theta} + \hat{\mathrm{n}}_z \sin\hat{\theta} & \hat{\mathrm{n}}_y\hat{\mathrm{n}}_y \mathrm{ver}\,\hat{\theta} + \cos\hat{\theta} & \hat{\mathrm{n}}_z\hat{\mathrm{n}}_y \mathrm{ver}\,\hat{\theta} - \hat{\mathrm{n}}_x \sin\hat{\theta} \\ \hat{\mathrm{n}}_x\hat{\mathrm{n}}_z \mathrm{ver}\,\hat{\theta} - \hat{\mathrm{n}}_y \sin\hat{\theta} & \hat{\mathrm{n}}_y\hat{\mathrm{n}}_z \mathrm{ver}\,\hat{\theta} + \hat{\mathrm{n}}_x \sin\hat{\theta} & \hat{\mathrm{n}}_z\hat{\mathrm{n}}_z \mathrm{ver}\,\hat{\theta} + \cos\hat{\theta} \end{bmatrix} \quad (27)$$

where $\mathrm{ver}\,\hat{\theta} = 1 - \cos\hat{\theta}$, $\hat{\mathrm{n}}_x = \mathrm{n}_x + \epsilon \mathrm{n}_{0x}$, *etc.* See [Rooney,1978] for the fully expanded version of the matrix.

Dual orthogonal matrices have been applied to spatial kinematics in the work of [McCarthy,1987] and [Pennock,1985] and have proven to be somewhat cumbersome due to the rather frightening amount of algebra involved in computing products of such matrices. This, of course, is largely due to the fact that the matrix stores 18 parameters to represent the 6 independent parameters necessary to completely determine a spatial displacement. The dual matrix, however, takes on rather simple forms in the special cases of screw displacements along coordinate axes and may therefore prove convenient for some applications. Another advantage of this representation is its matrix organization, which lends itself well to automated (parallel) manipulation.

3.2.2 Dual Unit Quaternion

Quaternions were originally introduced by Hamilton in the early 1840's in the context of investigating vector quotients. A quaternion is a 4-tuple of the form

$$\begin{aligned} \mathbf{q} &= q_1 + iq_2 + jq_3 + kq_4 \\ &= (q_1, q_2, q_3, q_4) \end{aligned} \quad (28)$$

where $q_1, q_2, q_3, q_4 \in \mathbb{R}$ and i, j, k are algebraic units with the property

$$i^2 = j^2 = k^2 = ijk = -1 \quad (29)$$

Interpreting the algebraic units as the unit vectors of a right-handed orthogonal coordinate frame, we can write a quaternion as a sum of a scalar and a vector, *i.e.*,

$$\mathbf{q} = q_1 + \mathbf{v} \qquad ; \qquad \mathbf{v} = iq_2 + jq_3 + kq_4 \tag{30}$$

It can be shown ([Horn,1987],[Funda,1988]) that the quaternion $\mathbf{q} = (\cos\frac{\theta}{2} + \mathbf{n}\sin\frac{\theta}{2})$, when applied to an arbitrary vector $\mathbf{r}$ in the form of a similarity transformation, effects a fixed-point rotation through an angle of θ about the spatial axis $\mathbf{n}$, *i.e.*,

$$\begin{aligned} \mathbf{r}' &= (\cos\frac{\theta}{2} + \mathbf{n}\sin\frac{\theta}{2}) * \mathbf{r} * (\cos\frac{\theta}{2} - \mathbf{n}\sin\frac{\theta}{2}) \\ &= \mathbf{Q}(\mathbf{n},\theta) * \mathbf{r} * \mathbf{Q}(\mathbf{n},\theta)^{-1} \end{aligned} \tag{31}$$

where $\mathbf{r}'$ is the rotated vector. However, in the present context, our concern is with rotating lines, rather than point vectors. This lead Clifford ([Clifford,1873]) to introduce *dual quaternions* (originally termed *biquaternions*). As expected, a dual quaternion is defined as a 4-tuple of dual numbers, *i.e.*,

$$\begin{aligned} \hat{\mathbf{q}} &= (q_1 + \epsilon q_{01}) + i(q_2 + \epsilon q_{02}) + j(q_3 + \epsilon q_{03}) + k(q_4 + \epsilon q_{04}) \\ &= (q_1 + \epsilon q_{01}, q_2 + \epsilon q_{02}, q_3 + \epsilon q_{03}, q_4 + \epsilon q_{04}) \\ &= (q_1, q_2, q_3, q_4) + \epsilon(q_{01}, q_{02}, q_{03}, q_{04}) \\ &= \mathbf{q} + \epsilon\mathbf{q}_0 \end{aligned} \tag{32}$$

where $\mathbf{q}$ and $\mathbf{q}_0$ are real quaternions. Addition and multiplication, as well as inverse, conjugate and norm are all defined analogously to the case of real quaternions ([Hamilton,1869]).

Applying the *Principle of Transference* again, we obtain (from Eq.(31)) the following expression for a general screw displacement of a dual vector $\hat{\mathbf{l}}$ through the dual angle $\hat{\theta} = \theta + \epsilon d$ along the spatial axis $\hat{\mathbf{n}}$:

$$\begin{aligned} \hat{\mathbf{l}}' &= (\cos\frac{\hat{\theta}}{2} + \hat{\mathbf{n}}\sin\frac{\hat{\theta}}{2}) * \hat{\mathbf{l}} * (\cos\frac{\hat{\theta}}{2} - \hat{\mathbf{n}}\sin\frac{\hat{\theta}}{2}) \\ &= \hat{\mathbf{Q}}(\hat{\mathbf{n}},\hat{\theta}) * \hat{\mathbf{l}} * \hat{\mathbf{Q}}(\hat{\mathbf{n}},\hat{\theta})^{-1} \end{aligned} \tag{33}$$

Note that this formulation necessitates the introduction of *half dual angles*. Moreover, $\hat{\mathbf{n}}$ in the above equation must be a *unit line vector* to ensure that the screw operator $\hat{\mathbf{Q}}$ is of unit magnitude, *i.e.*,

$$|\hat{\mathbf{Q}}| = (q_1 + \epsilon q_{01})^2 + (q_2 + \epsilon q_{02})^2 + (q_3 + \epsilon q_{03})^2 + (q_4 + \epsilon q_{04})^2 = 1 \tag{34}$$

This requirement translates into two distinct conditions

$$\begin{aligned} q_1^2 + q_2^2 + q_3^2 + q_4^2 &= 1 \\ q_1 q_{01} + q_2 q_{02} + q_3 q_{03} + q_4 q_{04} &= 0 \end{aligned} \tag{35}$$

which, imposed on the 8 parameters of a general dual quaternion, effectively reduce the number of d.o.f. of its parameter space to 6, as required.

The main advantages of the dual quaternion formulation of screw displacements are in the simplicity and compactness of the representation. It may also be the most intuitive of the representations, as the dual angular displacement and the unit line vector of the screw axis appear explicitly in its specification. Dual quaternions have been applied to spatial kinematics in the work of [Yang,1964].

3.2.3 Dual Special Unitary 2 x 2 Matrix

Dual unitary 2 x 2 matrices are obtained by applying the *Principle of Transference* to their real unitary matrix counterparts ([Rooney,1977]). A general dual unitary 2×2 matrix representing a spatial screw displacement is of the form ([Rooney,1978]):

$$\hat{\mathbf{D}} = \begin{bmatrix} \hat{a}_{11} + i\hat{b}_{11} & \hat{a}_{12} + i\hat{b}_{12} \\ -\hat{a}_{12} + i\hat{b}_{12} & \hat{a}_{11} - i\hat{b}_{11} \end{bmatrix} \tag{36}$$

where $\hat{a}_{ij} = a_{ij} + \epsilon a_{0ij}$ and $\hat{b}_{ij} = b_{ij} + \epsilon b_{0ij}$. As the matrix $\hat{\mathbf{D}}$ contains 8 parameters, the following restriction is imposed on the parameter space

$$\hat{a}_{11}^2 + \hat{b}_{11}^2 + \hat{a}_{12}^2 + \hat{b}_{12}^2 = 1 \tag{37}$$

The real and dual components of the above equation provide the two necessary conditions to reduce the number of independent parameters to 6.

A representation of a line $\hat{\mathbf{l}} = \langle \hat{l}_x, \hat{l}_y, \hat{l}_z \rangle$, such that its format is compatible with the dual unitary matrix, is obtained by using a *dual Hermitian matrix* of the form ([Rooney,1978])

$$\hat{\mathbf{l}} = \begin{bmatrix} \hat{l}_z & \hat{l}_x - i\hat{l}_y \\ \hat{l}_x + i\hat{l}_y & -\hat{l}_z \end{bmatrix} \tag{38}$$

A screw displacement is then performed by operating on the line matrix $\hat{\mathbf{l}}$ using a similarity transformation

$$\hat{\mathbf{l}}' = \hat{\mathbf{D}} * \hat{\mathbf{l}} * \hat{\mathbf{D}}^{-1} \tag{39}$$

where $\hat{\mathbf{D}}^{-1}$ is the matrix inverse of $\hat{\mathbf{D}}$ (Eq.(36)). The explicit form of the matrix $\hat{\mathbf{D}}$ representing a screw displacement along a spatial axis $\hat{\mathbf{n}} = \langle \hat{n}_x, \hat{n}_y, \hat{n}_z \rangle$ through a dual angle $\hat{\theta} = \theta + \epsilon d$ can be shown to be ([Rooney,1978]):

$$\hat{\mathbf{D}}(\hat{\mathbf{n}}, \hat{\theta}) = \begin{bmatrix} \cos\frac{\hat{\theta}}{2} + i\hat{n}_z \sin\frac{\hat{\theta}}{2} & \hat{n}_y \sin\frac{\hat{\theta}}{2} + i\hat{n}_x \sin\frac{\hat{\theta}}{2} \\ -\hat{n}_y \sin\frac{\hat{\theta}}{2} + i\hat{n}_x \sin\frac{\hat{\theta}}{2} & \cos\frac{\hat{\theta}}{2} - i\hat{n}_z \sin\frac{\hat{\theta}}{2} \end{bmatrix} \tag{40}$$

As in the case of dual quaternions, an important advantage of the dual special unitary 2×2 matrix representation is its conciseness. Recognizing that composition of successive spatial displacements using this approach yields much more elegant and compact algebraic expressions than the perhaps more intuitive 3×3 dual orthogonal matrix method, Denavit ([Denavit,1955]) used dual special unitary matrices for kinematic analysis of spatial linkages.

3.2.4 Dual Pauli Spin Matrices

This representation is closely related to the dual special unitary 2×2 matrix of Section 3.2.3, and is arrived at by considering the basic Pauli spin matrices ([Rooney,1978]):

$$\sigma_{\mathbf{x}} = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \qquad \sigma_{\mathbf{y}} = \begin{bmatrix} 0 & -i \\ i & 0 \end{bmatrix} \qquad \sigma_{\mathbf{z}} = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix} \tag{41}$$

Using these matrices, a line $\hat{\mathbf{l}} = \langle \hat{l}_x, \hat{l}_y, \hat{l}_z \rangle$ in space can be represented as

$$\hat{\mathbf{l}} = \hat{l}_x \sigma_{\mathbf{x}} + \hat{l}_y \sigma_{\mathbf{y}} + \hat{l}_z \sigma_{\mathbf{z}} \tag{42}$$

and a general screw displacement through a dual angle $\hat{\theta} = \theta + \epsilon d$ along the unit line vector $\hat{\mathbf{n}} = \langle \hat{n}_x, \hat{n}_y, \hat{n}_z \rangle$ is then expressed as a similarity transformation

$$\hat{\mathbf{l}}' = \hat{\mathbf{P}}(\hat{\mathbf{n}}, \hat{\theta}) * \hat{\mathbf{l}} * \hat{\mathbf{P}}(\hat{\mathbf{n}}, \hat{\theta})^{-1} \tag{43}$$

where the operator $\hat{\mathbf{P}}(\hat{\mathbf{n}}, \hat{\theta})$ is given by

$$\hat{\mathbf{P}}(\hat{\mathbf{n}}, \hat{\theta}) = (\cos\frac{\hat{\theta}}{2}\mathbf{I} + i \sin\frac{\hat{\theta}}{2}(\hat{n}_x \sigma_{\mathbf{x}} + \hat{n}_y \sigma_{\mathbf{y}} + \hat{n}_z \sigma_{\mathbf{z}})) \tag{44}$$

The symbol $\mathbf{I}$ above denotes the 2×2 identity matrix, and $\hat{\theta} = \theta + \epsilon d$, $\hat{\mathbf{n}} = \langle n_x + \epsilon n_{0x}, n_y + \epsilon n_{0y}, n_z + \epsilon n_{0z} \rangle$, as before.

Notice the similarity between this representation and the dual quaternion operator of Eq.(33). Moreover, expanding Eq.(44) into the corresponding matrix format yields precisely the dual special unitary 2×2 matrix of Eq.(36). Pauli spin matrices thus essentially provide a compact and elegant notation for dual special unitary 2×2 matrices, rather than a new representational formalism.

4 Computational Analysis of Line Transformations

From the standpoint of evaluating relative advantages of different representational schemes, we need to be concerned with the *compactness* of a given representation, as well as with its *computational efficiency* in performing basic spatial operations. Storage considerations are necessitated by the fact that on most existing hardware the cost (measured in terms of the number of required CPU cycles) of fetching an operand from memory exceeds the cost of performing a basic arithmetic operations. Consequently, an advantage of a particular representation in terms of storage may well outweigh a small disadvantage in terms of computational expense.

In the remainder of this section I will present my results of evaluating the computational behavior of the four representations of general screw displacements introduced in Section 3. The comparison will be done in terms of two fundamental operations: (1) *screw displacement of a line*, and (2) *composition of two screw displacement operators*. The costs for both *sequential* and *parallel* executions of the corresponding algorithms will be derived. To ensure correct interpretation of the stated results, a few comments about the way in which the numbers were obtained may be appropriate. This is particularly critical for the case of parallel algorithm cost analyses.

Costs for parallel algorithms are given in terms of the number of *processing elements* (PEs) and the number of *machine cycles* (cyc.) needed to perform the computation. A cycle, in this context, is defined as the time required by the hardware to execute a basic floating point add/subtract or multiply/divide operation. Parallel computations are structured so as to minimize the necesary number of cycles (*i.e.*, maximize computational concurrency) without regard to the number of PEs needed to support this optimal degree of parallelism.

4.1 Costs of Basic Operations on Dual Quantities

In order to facilitate speedier and more convenient cost evaluation in the upcoming sections, let us briefly summarize the costs of some basic operations on dual numbers and dual vectors.

Costs Of Basic Operations On Dual Quantities

		Sequential		Parallel	
		*	+	PEs	cyc.
dual number sum	$\hat{a} + \hat{b} = (a + b) + \epsilon(a_0 + b_0)$	0	2	2	1
dual number product	$\hat{a}\hat{b} = ab + \epsilon(a_0 b + a b_0)$	3	1	2	2
dual vector scaling	$\hat{s}\hat{\mathbf{n}} = (s + \epsilon s_0)\langle \hat{\mathrm{n}}_x, \hat{\mathrm{n}}_y, \hat{\mathrm{n}}_z \rangle$	9	3	6	2
dual dot product	$\hat{\mathbf{a}} \cdot \hat{\mathbf{b}} = \mathbf{a} \cdot \mathbf{b} + \epsilon(\mathbf{a} \cdot \mathbf{b}_0 + \mathbf{a}_0 \cdot \mathbf{b})$	9	7	6	4
dual cross product	$\hat{\mathbf{a}} \times \hat{\mathbf{b}} = \mathbf{a} \times \mathbf{b} + \epsilon(\mathbf{a} \times \mathbf{b}_0 + \mathbf{a}_0 \times \mathbf{b})$	18	12	12	3

The stated parallel costs for the above operations can be easily verified as optimal. Consequently, I will omit listing the explicit procedures.

4.2 Sequential Algorithms

4.2.1 Dual Orthogonal 3 × 3 Matrix

As we have seen in Section 3.2.1, the dual orthogonal 3 × 3 matrix is a highly redundant representation of a screw displacement, containing 18 parameters, of which only 6 are independent. In view of the considerations mentioned in the introductory paragraphs of this section, we will exploit the fact that the three dual column vectors $\hat{\mathbf{n}}, \hat{\mathbf{o}}, \hat{\mathbf{a}}$ of the matrix in Eq.(22) are mutually orthogonal, and will thus store only two of them (*e.g.*, $\hat{\mathbf{o}}$ and $\hat{\mathbf{a}}$). Note that this convention necessitates restoring the normal dual vector $\hat{\mathbf{n}}$ when the full matrix is needed for a particular operation. As we will see shortly, this represents pure overhead (one dual cross product) in the case of a screw transformation of a line, but translates into a significant saving (3 dual dot products) during composition of screw displacements, as the destination vector $\hat{\mathbf{n}}$ of the new matrix need not be computed. I feel that this, coupled with a 33% reduction in storage space, provides ample justification to adopt the proposed convention.

Let us now turn to the computational aspects of the two screw operations under consideration.

Screw Displacement: Labeling the rows of the dual matrix of Eq.(22) with the dual vectors $\hat{\mathbf{x}}, \hat{\mathbf{y}}$ and $\hat{\mathbf{z}}$, respectively, we can write the general screw displacement equation as follows

$$\hat{\mathbf{l}}' = \hat{\mathbf{R}} * \hat{\mathbf{l}} = \begin{bmatrix} \hat{\mathrm{n}}_x & \hat{\mathrm{o}}_x & \hat{\mathrm{a}}_x \\ \hat{\mathrm{n}}_y & \hat{\mathrm{o}}_y & \hat{\mathrm{a}}_y \\ \hat{\mathrm{n}}_z & \hat{\mathrm{o}}_z & \hat{\mathrm{a}}_z \end{bmatrix} \begin{bmatrix} \hat{\mathrm{l}}_x \\ \hat{\mathrm{l}}_y \\ \hat{\mathrm{l}}_z \end{bmatrix} = \begin{bmatrix} \hat{\mathbf{x}} \cdot \hat{\mathbf{l}} \\ \hat{\mathbf{y}} \cdot \hat{\mathbf{l}} \\ \hat{\mathbf{z}} \cdot \hat{\mathbf{l}} \end{bmatrix} \tag{45}$$

which requires 1 dual cross product to restore $\hat{\mathbf{n}}$ of the matrix $(18*, 12+)$, and 3 dual dot products to compute the coordinates of the displaced line $\hat{\mathbf{l}}'$ $(27*, 21+)$, for a total of $45*$ and $33+$.

Composition of Screw Displacements: Using a similar nomenclature as above, we can express the composition of two screw displacements as follows

$$\hat{\mathbf{R}} * \hat{\mathbf{R}}' = \begin{bmatrix} \hat{\mathrm{n}}_x & \hat{\mathrm{o}}_x & \hat{\mathrm{a}}_x \\ \hat{\mathrm{n}}_y & \hat{\mathrm{o}}_y & \hat{\mathrm{a}}_y \\ \hat{\mathrm{n}}_z & \hat{\mathrm{o}}_z & \hat{\mathrm{a}}_z \end{bmatrix} \begin{bmatrix} \hat{\mathrm{n}}'_x & \hat{\mathrm{o}}'_x & \hat{\mathrm{a}}'_x \\ \hat{\mathrm{n}}'_y & \hat{\mathrm{o}}'_y & \hat{\mathrm{a}}'_y \\ \hat{\mathrm{n}}'_z & \hat{\mathrm{o}}'_z & \hat{\mathrm{a}}'_z \end{bmatrix} = \begin{bmatrix} ? & \hat{\mathbf{x}} \cdot \hat{\mathbf{o}}' & \hat{\mathbf{x}} \cdot \hat{\mathbf{a}}' \\ ? & \hat{\mathbf{y}} \cdot \hat{\mathbf{o}}' & \hat{\mathbf{y}} \cdot \hat{\mathbf{a}}' \\ ? & \hat{\mathbf{z}} \cdot \hat{\mathbf{o}}' & \hat{\mathbf{z}} \cdot \hat{\mathbf{a}}' \end{bmatrix} \tag{46}$$

where the destination dual normal vector $\hat{\mathbf{n}}_d$ need not be computed, because it is (under the adopted convention) not explicitly stored. Moreover, since the source dual normal vector $\hat{\mathbf{n}}'$ only participates in the construction of $\hat{\mathbf{n}}_d$, it, too, need not be computed (*i.e.*, restored). Consequently, the cost of composing two screw transformations reduces to 1 dual cross product to restore $\hat{\mathbf{n}}$ $(18*, 12+)$, and 6 dual dot products $(54*, 42+)$ to compute the destination orientation and approach dual vectors $\hat{\mathbf{o}}_d$ and $\hat{\mathbf{a}}_d$. The total cost, then, is $72*$ and $54+$.

4.2.2 Dual Unit Quaternion

A dual unit quaternion is a 4-tuple of dual values and thus uses 8 numbers to encode the 6 independent displacement parameters. Due to minimal representational redundancy, no effort is made to further compact the representation and a dual unit quaternion screw operator is stored in a straightforward manner as 8 discrete values.

Screw Displacement: Recall from Section 3.2.2 that a real quaternion can be written as a sum of a scalar and a vector (Eq.(30)). Therefore, we can interpret a vector as a quaternion whose scalar part is zero. Likewise, we can treat dual vectors as dual quaternions with null (dual) scalar parts. Armed with this convention, recall that a real quaternion rotation of a point vector $\mathbf{r}$ through the quaternion $(s + \mathbf{u})$ can be computed as follows ([Horn,1987])

$$\begin{aligned} \mathbf{r}' &= \mathbf{q} * \mathbf{r} * \mathbf{q}^{-1} \\ &= (s + \mathbf{u}) * (0 + \mathbf{r}) * (s - \mathbf{u}) \\ &= \mathbf{r} + 2s(\mathbf{u} \times \mathbf{r}) + 2\mathbf{u} \times (\mathbf{u} \times \mathbf{r}) \end{aligned} \tag{47}$$

Applying the *Principle of Transference* once more, we can thus compute the screw displacement of Eq.(33) as

$$\begin{aligned} \hat{\mathbf{l}}' &= \hat{\mathbf{Q}}(\hat{\mathbf{n}}, \hat{\theta}) * \hat{\mathbf{l}} * \hat{\mathbf{Q}}(\hat{\mathbf{n}}, \hat{\theta})^{-1} \\ &= \hat{\mathbf{l}} + 2\hat{s}(\hat{\mathbf{u}} \times \hat{\mathbf{l}}) + 2\hat{\mathbf{u}} \times (\hat{\mathbf{u}} \times \hat{\mathbf{l}}) \end{aligned} \tag{48}$$

where $\hat{s} = \cos\frac{\hat{\theta}}{2}$ and $\hat{\mathbf{u}} = \hat{\mathbf{n}} \sin\frac{\hat{\theta}}{2}$. The cost of effecting a screw displacement using a dual unit quaternion then is 2 dual cross products $(36*, 24+)$, 1 dual vector scaling operation $(9*, 3+)$, and 2 dual vector additions $(0*, 12+)$, for a total of $45*$ and $39+$.[5]

Composition of Screw Displacements: By analogy to the real quaternion case ([Hamilton,1869]), we can again deduce the dual quaternion multiplication rule by resorting to the *Principle of Transference*, which gives

$$\begin{aligned} \hat{\mathbf{Q}} * \hat{\mathbf{Q}}' &= (\hat{s} + \hat{\mathbf{u}}) * (\hat{s}' + \hat{\mathbf{u}}') \\ &= (\hat{s}\hat{s}' - \hat{\mathbf{u}} \cdot \hat{\mathbf{u}}') + (\hat{s}\hat{\mathbf{u}}' + \hat{s}'\hat{\mathbf{u}} + \hat{\mathbf{u}} \times \hat{\mathbf{u}}') \end{aligned} \tag{49}$$

The list of costs in this case is somewhat more diverse — we have 1 dual number add $(0*, 2+)$, 1 dual number product $(3*, 1+)$, 2 scalings of a dual vector $(18*, 6+)$, 2 dual vector adds $(0*, 12+)$, 1 dual dot product $(9*, 7+)$, and 1 dual cross product $(18*, 12+)$. Adding all the numbers yields the total cost of $48*$ and $40+$.

[5] Multiplications by 2 can be implemented as exponent shifts and will therefore be neglected in this and all subsequent cost analyses.

4.2.3 Dual Special Unitary 2 × 2 Matrix

Referring back to Eq.(36) we see that 8 parameters completely define a dual special unitary 2 × 2 matrix. Again, as in the case of the dual unit quaternion operator, we will store all 8 parameters without attempting to further compress the representation.

Screw Displacement: Expanding the equation of a general screw displacement using dual special unitary 2 × 2 matrices (Eq.(39)) gives

$$\hat{\mathbf{l}}' = \hat{\mathbf{D}} * \hat{\mathbf{l}} * \hat{\mathbf{D}}^{-1} = \begin{bmatrix} \hat{a}_{11} + i\hat{b}_{11} & \hat{a}_{12} + i\hat{b}_{12} \\ -\hat{a}_{12} + i\hat{b}_{12} & \hat{a}_{11} - i\hat{b}_{11} \end{bmatrix} \begin{bmatrix} \hat{\mathrm{l}}_z & \hat{\mathrm{l}}_x - i\hat{\mathrm{l}}_y \\ \hat{\mathrm{l}}_x + i\hat{\mathrm{l}}_y & -\hat{\mathrm{l}}_z \end{bmatrix} \begin{bmatrix} \hat{a}_{11} - i\hat{b}_{11} & -\hat{a}_{12} - i\hat{b}_{12} \\ \hat{a}_{12} - i\hat{b}_{12} & \hat{a}_{11} + i\hat{b}_{11} \end{bmatrix} \tag{50}$$

Analyzing the cost of the product of the leftmost two matrices first, we see that each element of the resulting matrix requires 18∗ and 14+, for a total subproduct cost of 72∗ and 56+. Similarly, we find that the remaining matrix product accounts for 96∗ and 80+, bringing the the total to 168∗ and 136+.

Composition of Screw Displacements: Composing two screw displacements represented as dual special unitary 2 × 2 matrices corresponds to multiplying two 2 × 2 matrices with *dual complex* entries (Eq.(40)). This is precisely what we have done in the second phase of the screw transformation above and hence the cost of composing two successive screw displacements is 96∗ and 80+.

4.2.4 Dual Pauli Spin Matrices

The dual Pauli screw operator (Eq.(44)) is defined using 8 parameters. As with dual unit quaternions and dual special unitary 2 × 2 matrices, all 8 parameters are stored.

Screw Displacement: A general screw displacement of a line using the dual Pauli spin matrix operator is given by Eq.(43). Therefore, a numeric Pauli screw operator has the form (see Eq.(44))

$$\hat{\mathbf{P}}(\hat{\mathbf{n}}, \hat{\theta}) = (\hat{a}_1 \mathbf{I} + i\hat{a}_2 \sigma_{\mathbf{x}} + i\hat{a}_3 \sigma_{\mathbf{y}} + i\hat{a}_4 \sigma_{\mathbf{z}}) \tag{51}$$

where $\hat{a}_1$, $\hat{a}_2$, $\hat{a}_3$, and $\hat{a}_4$ represent the numeric dual values of $\cos\frac{\hat{\theta}}{2}$, $\sin\frac{\hat{\theta}}{2}\hat{n}_x$, $\sin\frac{\hat{\theta}}{2}\hat{n}_y$, and $\sin\frac{\hat{\theta}}{2}\hat{n}_z$, respectively. The inverse Pauli operator is obtained by negating the imaginary components in Eq.(51). Exploiting the properties of the basic Pauli matrices (Eq.(41)), *i.e.*,

$$\begin{aligned} \sigma_{\mathbf{x}}{}^2 = \sigma_{\mathbf{y}}{}^2 = \sigma_{\mathbf{z}}{}^2 = \mathbf{I} \\ \sigma_{\mathbf{x}}\sigma_{\mathbf{y}} = i\sigma_{\mathbf{z}} = -\sigma_{\mathbf{y}}\sigma_{\mathbf{x}} \\ \sigma_{\mathbf{y}}\sigma_{\mathbf{z}} = i\sigma_{\mathbf{x}} = -\sigma_{\mathbf{z}}\sigma_{\mathbf{y}} \\ \sigma_{\mathbf{z}}\sigma_{\mathbf{x}} = i\sigma_{\mathbf{y}} = -\sigma_{\mathbf{x}}\sigma_{\mathbf{z}} \end{aligned} \tag{52}$$

we can now derive the expression for the screw displaced line $\hat{\mathbf{l}}'$. A laborious derivation yields the following result

$$\begin{aligned} \hat{\mathbf{l}}' &= \hat{\mathbf{P}}(\hat{\mathbf{n}},\hat{\theta}) * \hat{\mathbf{l}} * \hat{\mathbf{P}}(\hat{\mathbf{n}},\hat{\theta})^{-1} \\ &= [(\hat{a}_1^2 + \hat{a}_2^2 - \hat{a}_3^2 - \hat{a}_4^2)\hat{l}_x + 2(\hat{a}_1\hat{a}_4 + \hat{a}_2\hat{a}_3)\hat{l}_y + 2(\hat{a}_2\hat{a}_4 - \hat{a}_1\hat{a}_3)\hat{l}_z]\sigma_{\mathbf{x}} \\ &\quad [2(\hat{a}_2\hat{a}_3 - \hat{a}_1\hat{a}_4)\hat{l}_x + (\hat{a}_1^2 - \hat{a}_2^2 + \hat{a}_3^2 - \hat{a}_4^2)\hat{l}_y + 2(\hat{a}_1\hat{a}_2 + \hat{a}_3\hat{a}_4)\hat{l}_z]\sigma_{\mathbf{y}} \\ &\quad [2(\hat{a}_1\hat{a}_3 + \hat{a}_2\hat{a}_4)\hat{l}_x + 2(\hat{a}_3\hat{a}_4 - \hat{a}_1\hat{a}_2)\hat{l}_y + (\hat{a}_1^2 - \hat{a}_2^2 - \hat{a}_3^2 + \hat{a}_4^2)\hat{l}_z]\sigma_{\mathbf{z}} \end{aligned} \tag{53}$$

Examining Eq.(53), we see that we need to compute 10 dual number products of the form $\hat{a}_i\hat{a}_j$, where $1 \leq i \leq j \leq 4$, which requires $30*, 10+$. Each of the 6 off-diagonal elements can be then computed using 1 dual number add/multiply pair for a total of $18*, 18+$. By reusing partial differences, the 3 diagonal elements can be evaluated with an additional $9*, 17+$. Finally, the three elements of each σ-component can be added together using 2 dual adds per component for a total cost contribution of $0*, 12+$. A general screw displacement of a line using the dual Pauli spin matrix operator therefore requires $57*$ and $57+$.

Composition of Screw Displacements: A derivation, similar to the one above, reveals the following form of a product of two dual Pauli screw operators

$$\begin{aligned} \hat{\mathbf{P}} * \hat{\mathbf{P}}' = \; & [(\hat{a}_1\hat{a}'_1 - \hat{a}_2\hat{a}'_2 - \hat{a}_3\hat{a}'_3 - \hat{a}_4\hat{a}'_4)\mathbf{I} + \\ & i(\hat{a}_2\hat{a}'_1 + \hat{a}_1\hat{a}'_2 + \hat{a}_4\hat{a}'_3 - \hat{a}_3\hat{a}_4)\sigma_{\mathbf{x}} + \\ & i(\hat{a}_3\hat{a}'_1 + \hat{a}_1\hat{a}'_3 + \hat{a}_2\hat{a}'_4 - \hat{a}_4\hat{a}'_2)\sigma_{\mathbf{y}} + \\ & i(\hat{a}_4\hat{a}'_1 + \hat{a}_1\hat{a}'_4 + \hat{a}_3\hat{a}'_2 - \hat{a}_2\hat{a}'_3)\sigma_{\mathbf{z}}] \end{aligned} \tag{54}$$

In order to evaluate Eq.(54), we need to compute all 16 dual products of the form $\hat{a}_i\hat{a}'_j$ for $1 \leq i,j \leq 4$, which requires $48*, 16+$. The 4 components of the composite

dual Pauli operator are then obtained with an additional 12 dual adds (0*, 24+), which brings the total to 48* and 40+.

4.3 Parallel Algorithms

4.3.1 Dual Orthogonal 3 x 3 Matrix

Screw Displacement: Expanding Eq.(45) of Section 4.2.1 we obtain the following expression

$$\hat{\mathrm{l}}' = \begin{bmatrix} \mathrm{n}_x\mathrm{l}_x + \epsilon(\mathrm{n}_{0x}\mathrm{l}_x + \mathrm{n}_x\mathrm{l}_{0x}) + \mathrm{o}_x\mathrm{l}_y + \epsilon(\mathrm{o}_{0x}\mathrm{l}_y + \mathrm{o}_x\mathrm{l}_{0y}) + \mathrm{a}_x\mathrm{l}_z + \epsilon(\mathrm{a}_{0x}\mathrm{l}_z + \mathrm{a}_x\mathrm{l}_{0z}) \\ \mathrm{n}_y\mathrm{l}_x + \epsilon(\mathrm{n}_{0y}\mathrm{l}_x + \mathrm{n}_y\mathrm{l}_{0x}) + \mathrm{o}_y\mathrm{l}_y + \epsilon(\mathrm{o}_{0y}\mathrm{l}_y + \mathrm{o}_y\mathrm{l}_{0y}) + \mathrm{a}_y\mathrm{l}_z + \epsilon(\mathrm{a}_{0y}\mathrm{l}_z + \mathrm{a}_y\mathrm{l}_{0z}) \\ \mathrm{n}_z\mathrm{l}_x + \epsilon(\mathrm{n}_{0z}\mathrm{l}_x + \mathrm{n}_z\mathrm{l}_{0x}) + \mathrm{o}_z\mathrm{l}_y + \epsilon(\mathrm{o}_{0z}\mathrm{l}_y + \mathrm{o}_z\mathrm{l}_{0y}) + \mathrm{a}_z\mathrm{l}_z + \epsilon(\mathrm{a}_{0z}\mathrm{l}_z + \mathrm{a}_z\mathrm{l}_{0z}) \end{bmatrix}$$

A parallel procedure to carry out the above computation is outlined below. The numeric column labels refer to the 6 columns of additive terms in the above equation.

Screw Displacement – 3 x 3 Matrix		
1.	12*	dual cross product – phase 1 ($\hat{\mathbf{n}}$)
	6*	dual part products involving $\hat{\mathbf{o}}$ (col. 4)
2.	6*,6+	dual cross product – phase 2 ($\hat{\mathbf{n}}$)
	6*	dual part products involving $\hat{\mathbf{a}}$ (col. 6)
3.	6+	dual cross product – phase 3 ($\hat{\mathbf{n}}$)
	6*	real part products involving $\hat{\mathbf{o}}$,$\hat{\mathbf{a}}$ (cols. 3,5)
	6+	dual part sums involving $\hat{\mathbf{o}}$,$\hat{\mathbf{a}}$ (cols. 4,6)
4.	9*	products involving $\hat{\mathbf{n}}$ (cols. 1,2)
	6+	sum up real parts / dual parts involving $\hat{\mathbf{o}}$, $\hat{\mathbf{a}}$ (cols. 3,4,5,6)
5.	3+	dual part sums involving $\hat{\mathbf{n}}$ (col. 2)
6.	3+	compute real part of result vector (add on col. 1)
	3+	compute dual part of result vector (add on col. 2)

requiring 6 cycles and 18 processing elements (PEs).

Composition of Screw Displacements: Referring back to Eq.(46), we see that we again need to restore the source normal dual vector $\hat{\mathbf{n}}$ and compute two dual dot products involving $\hat{\mathbf{n}}$. Exploiting the similarity of this procedure to the one above, we can proceed as follows

		Composition of Screws – 3 × 3 Matrix
1.	12∗	dual cross product – phase 1 ($\hat{\mathbf{n}}$)
	12∗	dual part products involving $\hat{\mathbf{o}}'$ but not $\hat{\mathbf{n}}$
2.	6∗,6+	dual cross product – phase 2 ($\hat{\mathbf{n}}$)
	12∗	dual part products involving $\hat{\mathbf{a}}'$ but not $\hat{\mathbf{n}}$
3.	6+	dual cross product – phase 3 ($\hat{\mathbf{n}}$)
	12∗	real part products involving $\hat{\mathbf{o}}'$,$\hat{\mathbf{a}}'$ but not $\hat{\mathbf{n}}$
	12+	dual part sums involving $\hat{\mathbf{o}}'$,$\hat{\mathbf{a}}'$ but not $\hat{\mathbf{n}}$
4.	18∗	products involving $\hat{\mathbf{n}}$
	6+	sum up real parts / dual parts involving $\hat{\mathbf{o}}'$ but not $\hat{\mathbf{n}}$
5.	6+	dual part sums involving $\hat{\mathbf{n}}$
	6+	sum up real parts / dual parts involving $\hat{\mathbf{a}}'$ but not $\hat{\mathbf{n}}$
6.	6+	compute real and dual parts of result vector involving $\hat{\mathbf{o}}'$
	6+	compute real and dual parts of result vector involving $\hat{\mathbf{a}}'$

This time we need 24 PEs, but still only 6 cycles.

4.3.2 Dual Unit Quaternion

Screw Displacement: Examining Eq.(48) of Section 4.2.2 we see that two dual cross products need to be carried out in succession, accounting for at leat 6 cycles. However, with some careful planning, we can evaluate the complete expression using only one additional cycle and a total of 18 PEs, as follows

Screw Displacement – Dual Quaternion		
1.	12*	dual cross product – phase 1 ($\hat{\mathbf{u}} \times \hat{\mathbf{l}}$)
2.	6*,6+	dual cross product – phase 2 ($\hat{\mathbf{u}} \times \hat{\mathbf{l}}$)
3.	6+	dual cross product – phase 3 ($\hat{\mathbf{u}} \times \hat{\mathbf{l}}$)
4.	12*	dual cross product – phase 1 ($\hat{\mathbf{u}} \times (\hat{\mathbf{u}} \times \hat{\mathbf{l}})$)
	6*	dual vector scaling – phase 1 ($\hat{s}(\hat{\mathbf{u}} \times \hat{\mathbf{l}})$)
5.	6*,6+	dual cross product – phase 2 ($\hat{\mathbf{u}} \times (\hat{\mathbf{u}} \times \hat{\mathbf{l}})$)
	3*,3+	dual vector scaling – phase 2 ($\hat{s}(\hat{\mathbf{u}} \times \hat{\mathbf{l}})$)
6.	6+	dual cross product – phase 3 ($\hat{\mathbf{u}} \times (\hat{\mathbf{u}} \times \hat{\mathbf{l}})$)
	6+	dual vector add ($\hat{\mathbf{e}} = \hat{\mathbf{l}} + 2\hat{s}(\hat{\mathbf{u}} \times \hat{\mathbf{l}})$)
7.	6+	final dual vector add ($\hat{\mathbf{l}}' = \hat{\mathbf{e}} + 2\hat{\mathbf{u}} \times (\hat{\mathbf{u}} \times \hat{\mathbf{l}})$)

Composition of Screw Displacements: A thorough analysis of Eq.(49) suggests the following parallel procedure

Composition of Screws – Dual Quaternion		
1.	12*	dual cross product – phase 1 ($\hat{\mathbf{u}} \times \hat{\mathbf{u}}'$)
	6*	dual dot product – phase 1 ($\hat{\mathbf{u}} \cdot \hat{\mathbf{u}}'$)
	6*	dual vector scaling – phase 1 ($\hat{s}\hat{\mathbf{u}}'$)
	6*	dual vector scaling – phase 1 ($\hat{s}'\hat{\mathbf{u}}$)
2.	6*,6+	dual cross product – phase 2 ($\hat{\mathbf{u}} \times \hat{\mathbf{u}}'$)
	3*,3+	dual dot product – phase 2 ($\hat{\mathbf{u}} \cdot \hat{\mathbf{u}}'$)
	3*,3+	dual vector scaling – phase 2 ($\hat{s}\hat{\mathbf{u}}'$)
	3*,3+	dual vector scaling – phase 2 ($\hat{s}'\hat{\mathbf{u}}$)
3.	6+	dual cross product – phase 3 ($\hat{\mathbf{u}} \times \hat{\mathbf{u}}'$)
	2+	dual dot product – phase 3 ($\hat{\mathbf{u}} \cdot \hat{\mathbf{u}}'$)
	2*	dual product – phase 1 ($\hat{s}\hat{s}'$)
4.	2+	dual dot product – phase 4 ($\hat{\mathbf{u}} \cdot \hat{\mathbf{u}}'$)
	1*,1+	dual product – phase 2 ($\hat{s}\hat{s}'$)
	6+	dual vector sum ($\hat{\mathbf{e}} = \hat{s}\hat{\mathbf{u}}' + \hat{s}'\hat{\mathbf{u}}$)
5.	2+	dual scalar result ($\hat{s}\hat{s}' - \hat{\mathbf{u}} \cdot \hat{\mathbf{u}}'$)
	6+	dual vector result ($\hat{\mathbf{e}} + (\hat{\mathbf{u}} \times \hat{\mathbf{u}}')$)

which requires 5 cycles and 30 PEs. Note that we could save two PEs by distributing the two scaling operations between the first 3 cycles (instead of the first 2). I have omitted this optimization in the procedure above for clarity — however, the cost of 5 cycles and 28 PEs has been entered in the summary tables in Section 4.4.

4.3.3 Dual Special Unitary 2×2 Matrix

Screw Displacement: To avoid computing redundant products, we will again (as in Section 4.2.3) evaluate the triple matrix product of Eq.(50) in two stages. We will compute the product of the leftmost two matrices in the first stage, and the remaining product in the second. Expanding Eq.(50), we obtain a matrix product of the form

$$\hat{\mathbf{I}}' = (\hat{\mathbf{D}} * \hat{\mathbf{I}}) * \hat{\mathbf{D}}^{-1} = \hat{\mathbf{M}} * \hat{\mathbf{D}}^{-1} =$$

$$\begin{bmatrix} (\hat{a}_{12}\hat{\mathbf{l}}_x - \hat{b}_{12}\hat{\mathbf{l}}_y + \hat{a}_{11}\hat{\mathbf{l}}_z) + i(\hat{b}_{12}\hat{\mathbf{l}}_x + \hat{a}_{12}\hat{\mathbf{l}}_y + \hat{b}_{11}\hat{\mathbf{l}}_z) & | \\ - & - \\ & | \end{bmatrix} \begin{bmatrix} \hat{a}_{11} - i\hat{b}_{11} & | \\ - - - - & - \\ & | \end{bmatrix}$$

where only one entry in each matrix is shown, the remaining three being of the same format in both cases. A parallel procedure, proceding in two stages as oulined above, can then be organized as follows

Screw Displacement – 2×2 Matrix		
1.	48*	24 dual products – phase 1 (6 per element of **M**)
2.	24*,24+	24 dual products – phase 2
3.	16+	8 dual sums (2 per element of **M**)
4.	16+	8 dual sums (2 per element of **M**) (**** stage 1 completed ****)
5.	64*	32 dual products – phase 1 (8 per element of **M**)
6.	32*,32+	32 dual products – phase 2
7.	32+	16 dual sums (4 per element of $\hat{\mathbf{I}}'$)
8.	16+	8 dual sums (2 per element of $\hat{\mathbf{I}}'$) (**** stage 2 completed ****)

Note that the above procedure requires 8 cycles and 64 PEs.

Composition of Screw Displacements: Composition of two screw displacements using dual special unitary 2×2 matrices corresponds to stage 2 of the above procedure, and hence requires 4 cycles and 64 PEs.

4.3.4 Dual Pauli Spin Matrices

Screw Displacement: Referring to Eq.(53) of Section 4.2.4 and proceding in a straightforward fashion, we arrive at the following parallel procedure

	Screw Displacement – Pauli Matrices	
1.	20∗	10 dual products – phase 1 ($\hat{a}_i\hat{a}_j$)
2.	10∗,10+	10 dual products – phase 2
3.	12+	6 dual sums (off-diagonal elements)
	8+	4 dual sums (diagonal elements)
4.	12∗	6 dual products – phase 1 (off-diagonal elements)
	6+	3 dual sums (diagonal elements)
5.	6∗,6+	6 dual products – phase 2 (off-diagonal elements)
	6∗	3 dual products – phase 1 (diagonal elements)
6.	3∗,3+	3 dual products – phase 2 (diagonal elements)
	6+	3 dual sums – sum up off-diagonal elements in each row
7.	6+	3 dual sums – add diagonal elements to the result vector

where (as in Section 4.2.4) partial differences have been reused in computing the diagonal element sums. The resources required for the above procedure are 7 cycles and 20 PEs.

Composition of Screw Displacements: Translating Eq.(54) into parallel code yields the following procedure

	Composition of Screws – Pauli Matrices	
1.	32∗	16 dual products – phase 1 ($\hat{a}_i\hat{a}_j$)
2.	16∗,16+	16 dual products – phase 2
3.	16+	8 dual sums (2 per element)
4.	8+	4 dual sums (1 per element)

requiring a total of 4 cycles and 32 PEs.

4.4 Summary

The following tables provide a summary of *storage requirements* and *computational performance* of the four representations of a spatial screw displacement described in Section 3.2.

Storage Requirements				
	$\hat{\mathbf{R}}$	$\hat{\mathbf{Q}}$	$\hat{\mathbf{D}}$	$\hat{\mathbf{P}}$
# of floats	12	8	8	8

Sequential Execution Of Screw Operations								
	$\hat{\mathbf{R}}$		$\hat{\mathbf{Q}}$		$\hat{\mathbf{D}}$		$\hat{\mathbf{P}}$	
	*	+	*	+	*	+	*	+
Screw Displacement	45	33	45	39	168	136	57	57
Composition of Screws	72	54	48	40	96	80	48	40

Parallel Execution Of Screw Operations								
	$\hat{\mathbf{R}}$		$\hat{\mathbf{Q}}$		$\hat{\mathbf{D}}$		$\hat{\mathbf{P}}$	
	PEs	cyc.	PEs	cyc.	PEs	cyc.	PEs	cyc.
Screw Displacement	18	6	18	7	64	8	20	7
Composition of Screws	24	6	28	5	64	4	32	4

5 Application — Robot Kinematics

The most basic problem in robot manipulator control is the positional (*i.e.*, kinematic) analysis of the robot linkage. The problem is normally treated in terms of two subproblems: *direct kinematics* and *inverse kinematics*.

The direct kinematics problem can be stated as follows: knowing the structural parameters of the manipulator (*e.g.*, link dimensions, *etc.*) and instantaneous values of all its joint variables (angles, displacements), compute the position and orientation of the manipulator's end-effector with respect to some designated reference coordinate frame. The solution is obtained in a straightforward fashion by multiplying out a chain of transformation operators relating successive coordinate frames associated with the links of the manipulator.

Conversely, and more interestingly, inverse kinematics addresses the opposite problem: given a desired position and orientation information for the end-effector, determine the value of the joint variable vector $(\vartheta_1, \vartheta_2, \ldots, \vartheta_n)$ that corresponds to the specified end-effector location.[6] The solution to this problem is required much more often in robot manipulator control than its direct kinematics counterpart and is essential in giving the manipulator the ability to perform tasks such as tracking a particular spatial trajectory. As we will see, solving inverse kinematics is also the more challenging of the two problems.

I will begin this section with a brief presentation of the classical mathematical model of open-loop spatial mechanisms (*i.e.*, robot manipulator linkages), based on Denavit-Hartenberg parameters ([Denavit,1955]). Having established the terminology and the necessary mathematical conventions, I will then outline two approaches to solving the inverse kinematics problem of a general n-d.o.f. robot manipulator, using the theory of spatial screw displacements presented in the previous sections. The first method utilizes dual 3×3 orthogonal matrices and represents a classical approach to the inverse kinematics problem using screws. My discussion of the procedure is based on a paper by Pennock and Yang ([Pennock,1985]). As an alternative approach, I will then propose a method, based on *orthogonal* dual unit quaternion screw operators. The development of that section represents an adaptation of the algorithm used by Yang and Freundestein ([Yang,1964]) in solving the positional

[6] I will in this paper use the term "location" to denote both position and orientation in space.

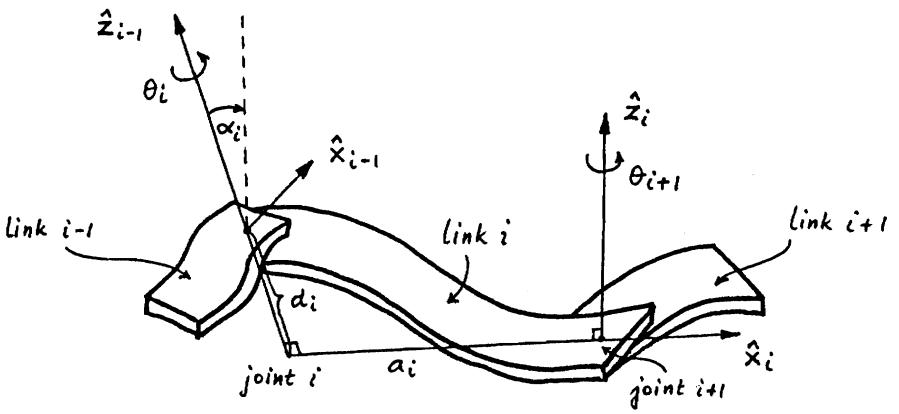

Figure 6: Denavit-Hartenberg link coordinate system parameters.

equations of a spatial four-link closed-loop mechanism.

The two approaches will be discussed in rather general terms, without explicit computations and results pertaining to specific manipulators. The interested reader is referred to the referenced papers for details of actual derivations.

5.1 Denavit-Hartenberg Parameters

A general-purpose robot manipulator is a spatial mechanism consisting of several rigid links, connected together in series by revolute and prismatic joints. With one end of the linkage fixed to a supporting base, the mechanism forms an open-loop chain, where each joint contributes one d.o.f. of movement in space. Links and joints are numbered outward from the base, so that joint 1 connects links 1 to the supporting base, and in general, joint i connects links i and $i-1$. In order to be able to assign coordinate frames to the manipulator linkage, we will define two unit line vectors per link, as follows (see Figure 6)

- unit line vector $\hat{\mathbf{z}}_{i-1}$, along the axis of joint i, and
- unit line vector $\hat{\mathbf{x}}_i$, coincident with the common normal between $\hat{\mathbf{z}}_{i-1}$ and $\hat{\mathbf{z}}_i$

The coordinate frame $\mathcal{F}_i$ associated with link i is now defined as the frame subtended by the unit line vectors $\hat{\mathbf{x}}_i$ and $\hat{\mathbf{z}}_i$. Moreover, we will refer to the coordinate frame defined by the dual vectors $\hat{\mathbf{x}}_i$ and $\hat{\mathbf{z}}_{i-1}$ as the joint i frame $\mathcal{J}_i$.

The Denavit-Hartenberg representation of a link and its relative position/orientation with respect to adjacent links thus depends on four geometric parameters, illustrated in Figure 6

1. θ_i - the joint angle measured from the $\hat{\mathbf{x}}_{i-1}$ axis to the $\hat{\mathbf{x}}_i$ axis about $\hat{\mathbf{z}}_{i-1}$ (using the right-hand rule)

2. d_i - the distance along $\hat{\mathbf{z}}_{i-1}$ from the origin of the link $i-1$ coordinate frame $\mathcal{F}_{i-1}$ to the origin of the joint i frame $\mathcal{J}_i$

3. α_i - the link twist angle measured from the $\hat{\mathbf{z}}_{i-1}$ axis to the $\hat{\mathbf{z}}_i$ axis about $\hat{\mathbf{x}}_i$ (using the right-hand rule)

4. a_i - the distance along $\hat{\mathbf{x}}_i$ from the origin of the joint frame $\mathcal{J}_i$ to the origin of the link frame $\mathcal{F}_i$

For a rotary joint, the parameters d_i, α_i and a_i are constant and θ_i is the joint variable that changes as the linkage moves. For a prismatic joint, on the other hand, the joint variable is d_i, while θ_i, α_i and a_i represent constant parameters of the link.

Referring to Figure 6 we see that the four parameters described above can be interpreted as a pair of dual angles. Grouping α_i and a_i together, we obtain a constant dual angle

$$\hat{\alpha}_i = \alpha_i + \epsilon a_i \tag{55}$$

between the adjacent joint axes $\hat{\mathbf{z}}_{i-1}$ and $\hat{\mathbf{z}}_i$, which completely describes the shape and proportions of the link i. The parameters α_i and a_i are referred to as the *twist angle* and the *length* of link i, respectively.

Similarly, the position of link i with respect to link $i-1$ is completely defined by the dual angle

$$\hat{\theta}_i = \theta_i + \epsilon d_i \tag{56}$$

between $\hat{\mathbf{x}}_{i-1}$ and $\hat{\mathbf{x}}_i$, with the positive sense in the direction of $\hat{\mathbf{z}}_{i-1}$.

We can therefore express the relative position and orientation of link i (*i.e.*, its associated frame $\mathcal{F}_i$) with respect to link $i-1$ (*i.e.*, the frame $\mathcal{F}_{i-1}$) as a composition of two successive screw displacements

$$\mathcal{F}_i = (\text{Screw}(\hat{\theta}_i)\text{Screw}(\hat{\alpha}_i))\mathcal{F}_{i-1} \tag{57}$$

The screw operator Screw($\hat{\theta}_i$) performs a rotation of the frame $\mathcal{F}_{i-1}$ about $\hat{\mathbf{z}}_{i-1}$ through an angle of θ_i (causing $\hat{\mathbf{x}}_{i-1}$ and $\hat{\mathbf{x}}_i$ to become parallel) and a linear displacement along $\hat{\mathbf{z}}_{i-1}$ through a distance d_i (making $\hat{\mathbf{x}}_{i-1}$ and $\hat{\mathbf{x}}_i$ coincident). The second screw displacement Screw($\hat{\alpha}_i$) then rotates the axis $\hat{\mathbf{z}}_{i-1}$ about $\hat{\mathbf{x}}_i$ through an angle of α_i, aligning it with $\hat{\mathbf{z}}_i$, and translates the frame $\mathcal{F}_{i-1}$ along $\hat{\mathbf{x}}_i$ a distance a_i, bringing the two frames into coincidence.

We will refer to the parenthesized screw product in Eq.(57) as the *link transformation operator* and will denote it by the symbol $\hat{\mathbf{A}}_i$.

5.2 Inverse Kinematics With Dual 3×3 Matrices

Using the mathematical conventions of Section 5.1 and opting to represent spatial screw displacements with dual 3×3 matrices, we can compute (from Eq.(57)) the general dual transformation matrix $\hat{\mathbf{A}}_i$ relating frames $\mathcal{F}_{i-1}$ and $\mathcal{F}_i$ as follows ([Pennock,1985])

$$\begin{aligned}\hat{\mathbf{A}}_i &= \begin{bmatrix} C\hat{\theta}_i & -S\hat{\theta}_i & 0 \\ S\hat{\theta}_i & C\hat{\theta}_i & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & C\hat{\alpha}_i & -S\hat{\alpha}_i \\ 0 & S\hat{\alpha}_i & C\hat{\alpha}_i \end{bmatrix} \\ &= \begin{bmatrix} C\hat{\theta}_i & -S\hat{\theta}_i C\hat{\alpha}_i & S\hat{\theta}_i S\hat{\alpha}_i \\ S\hat{\theta}_i & C\hat{\theta}_i C\hat{\alpha}_i & -C\hat{\theta}_i S\hat{\alpha}_i \\ 0 & S\hat{\alpha}_i & C\hat{\alpha}_i \end{bmatrix} \end{aligned} \tag{58}$$

where $S\hat{\theta}_i$, $C\hat{\theta}_i$, $S\hat{\alpha}_i$, $C\hat{\alpha}_i$ denote $\sin\hat{\theta}_i$, $\cos\hat{\theta}_i$, $\sin\hat{\alpha}_i$, and $\cos\hat{\alpha}_i$, respectively. By separating real and dual components, we can write a dual orthogonal transformation matrix $\hat{\mathbf{A}}_i$ as

$$\hat{\mathbf{A}}_i = \mathbf{A}_i + \epsilon\mathbf{A}_{0i} \tag{59}$$

where $\mathbf{A}_i$ is the real orientation matrix, giving the relative orientation of the frame $\mathcal{F}_i$ with respect to $\mathcal{F}_{i-1}$, and the dual part $\mathbf{A}_{0i}$ represents the moment matrix about the origin of $\mathcal{F}_{i-1}$ for the three axes of the link coordinate frame $\mathcal{F}_i$. Therefore, the dual part matrix $\mathbf{A}_{0i}$ can be written as the product

$$\mathbf{A}_{0i} = \mathbf{P}\mathbf{A}_i \tag{60}$$

where $\mathbf{P}$ is the skew symmetric cross-product moment matrix

$$\mathbf{P} = \begin{bmatrix} 0 & -p_z & p_y \\ p_z & 0 & -p_x \\ -p_y & p_x & 0 \end{bmatrix} \tag{61}$$

consisting of components of the vector $\mathbf{p}$, which gives the position of the origin of the frame $\mathcal{F}_i$ with respect to $\mathcal{F}_{i-1}$.

For an n-d.o.f. manipulator, we may then express the position and orientation of the terminal link (*i.e.*, frame $\mathcal{F}_n$) with respect to the base frame $\mathcal{F}_0$ as a product of successive dual matrix transformations

$$\hat{\mathbf{T}}_n = \hat{\mathbf{A}}_1 \hat{\mathbf{A}}_2 \cdots \hat{\mathbf{A}}_n \tag{62}$$

where each matrix $\hat{\mathbf{A}}_i$ is of the form of Eq.(58). Since 6 d.o.f. suffice for a manipulator to achieve an arbitrary position and orientation in the workspace, let us look briefly at the special case of the problem with $n = 6$.

According to the *Principle of Transference*, we can express the position and orientation of the end-effector with respect to the base frame as

$$\hat{\mathbf{T}}_6 = (\mathbf{I} + \epsilon \mathbf{P})\mathbf{T}_6 \tag{63}$$

where $\mathbf{I}$ is the 3×3 identity matrix, $\mathbf{P}$ is the skew-symmetric moment matrix of Eq.(61), and $\mathbf{T}_6$ is a 3×3 real orthogonal matrix, specifying the orientation of $\mathcal{F}_6$ with respect to $\mathcal{F}_0$. The inverse kinematics problem then becomes: given $\mathbf{P}$ and $\mathbf{T}_6$, compute the 6 joint variables ϑ_i, $1 \leq i \leq 6$, where $\vartheta_i = \theta_i$ if joint i is revolute, and $\vartheta_i = d_i$ if joint i is prismatic.

Substituting $\mathbf{A}_i + \epsilon \mathbf{A}_{0i}$ for $\hat{\mathbf{A}}_i$ in Eq.(62), expanding it for $n = 6$, and setting the expanded result equal to Eq.(63) yields the following pair of real and dual part equations

$$\mathbf{T}_6 = \mathbf{A}_1 \mathbf{A}_2 \cdots \mathbf{A}_6 \tag{64}$$

$$\begin{aligned} \mathbf{P}\mathbf{T}_6 = {} & \mathbf{A}_{01}\mathbf{A}_2\mathbf{A}_3\mathbf{A}_4\mathbf{A}_5\mathbf{A}_6 + \mathbf{A}_1\mathbf{A}_{02}\mathbf{A}_3\mathbf{A}_4\mathbf{A}_5\mathbf{A}_6 + \\ & \mathbf{A}_1\mathbf{A}_2\mathbf{A}_{03}\mathbf{A}_4\mathbf{A}_5\mathbf{A}_6 + \mathbf{A}_1\mathbf{A}_2\mathbf{A}_3\mathbf{A}_{04}\mathbf{A}_5\mathbf{A}_6 + \\ & \mathbf{A}_1\mathbf{A}_2\mathbf{A}_3\mathbf{A}_4\mathbf{A}_{05}\mathbf{A}_6 + \mathbf{A}_1\mathbf{A}_2\mathbf{A}_3\mathbf{A}_4\mathbf{A}_5\mathbf{A}_{06} \end{aligned} \tag{65}$$

Some rearranging of the above equation gives ([Pennock,1985])

$$\begin{aligned}\mathbf{A}_1^{-1}(\mathbf{PA}_1 - \mathbf{A}_{01}) &= \mathbf{A}_{02}\mathbf{A}_2^{-1} + \mathbf{A}_2(\mathbf{A}_{03}\mathbf{A}_3^{-1})\mathbf{A}_2^{-1} + \\ &\quad (\mathbf{A}_2\mathbf{A}_3)\mathbf{A}_{04}\mathbf{A}_4^{-1}(\mathbf{A}_2\mathbf{A}_3)^{-1} + \\ &\quad (\mathbf{A}_2\mathbf{A}_3\mathbf{A}_4)\mathbf{A}_{05}\mathbf{A}_5^{-1}(\mathbf{A}_2\mathbf{A}_3\mathbf{A}_4)^{-1} + \\ &\quad (\mathbf{A}_2\mathbf{A}_3\mathbf{A}_4\mathbf{A}_5)\mathbf{A}_{06}\mathbf{A}_6^{-1}(\mathbf{A}_2\mathbf{A}_3\mathbf{A}_4\mathbf{A}_5)^{-1} \qquad (66)\end{aligned}$$

where each term in the latter equation is a skew-symmetric matrix. We can further simplify Eq.(66) by exploiting the fact that for most robot manipulators, the twist double angle $\hat{\alpha}_6$ is zero. Therefore, we have $S\hat{\alpha}_6 = 0$, $C\hat{\alpha}_6 = 1$, and a bit of algebra reveals that

$$\mathbf{A}_{06} = \mathbf{PA}_6 = \mathbf{A}_6\mathbf{D} \qquad (67)$$

where $\mathbf{D}$ is the skew-symmetric matrix involving d_6, the dual part of $\hat{\theta}_6$

$$\mathbf{D} = \begin{bmatrix} 0 & -d_6 & 0 \\ d_6 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \qquad (68)$$

Finally, substituting Eq.(67) into the right-most term of Eq.(66), we obtain the final dual part equation ([Pennock,1985])

$$\begin{aligned}\mathbf{A}_1^{-1}(\mathbf{PA}_1 - \mathbf{T}_6\mathbf{D}\mathbf{T}_6^{-1}\mathbf{A}_1 - \mathbf{A}_{01}) &= \mathbf{A}_{02}\mathbf{A}_2^{-1} + \mathbf{A}_2(\mathbf{A}_{03}\mathbf{A}_3^{-1})\mathbf{A}_2^{-1} + \\ &\quad (\mathbf{A}_2\mathbf{A}_3)\mathbf{A}_{04}\mathbf{A}_4^{-1}(\mathbf{A}_2\mathbf{A}_3)^{-1} + \\ &\quad (\mathbf{A}_2\mathbf{A}_3\mathbf{A}_4)\mathbf{A}_{05}\mathbf{A}_5^{-1}(\mathbf{A}_2\mathbf{A}_3\mathbf{A}_4)^{-1} \qquad (69)\end{aligned}$$

The solution for the joint variable vector $(\vartheta_1, \vartheta_2, \ldots, \vartheta_6)$ can now be obtained systematically by comparing left and right hand sides of Eqs.(64,69). The interested reader is refrerred to [Pennock,1985] for an illustration of the method on three standard robot manipulators.

5.3 Inverse Kinematics With Orthogonal Dual Unit Quaternions

In this section I will outline an approach to solving the inverse kinematics problem for an arbitrary n-d.o.f. robot manipulator using *orthogonal dual unit quaternion screw operators*. The orthogonal dual quaternion screw operator represents a special

case of the corresponding general operator described in Section 3.2.2 (Eq.(33)) in that the screw axis is always *orthogonal* to the unit line vector being screw-displaced. I will first provide a brief derivation of the orthogonal operator and then turn to the outline of a procedure to solve the inverse kinematics problem.

Recall from Section 2.5 (Eq.(18)) that the algebraic product of two unit line vectors $\hat{\mathbf{a}}$ and $\hat{\mathbf{b}}$, whose relative position and orientation is defined by the dual angle $\hat{\alpha} = \alpha + \epsilon d$, is given by

$$\hat{\mathbf{a}}\hat{\mathbf{b}} = -\cos\hat{\alpha} + \hat{\mathbf{e}}\sin\hat{\alpha} \tag{70}$$

Therefore, when $\hat{\alpha} = 0°$ (*i.e.*, when the dual vectors $\hat{\mathbf{a}}$ and $\hat{\mathbf{b}}$ are coaxial) we have $\sin\hat{\alpha} = 0$, $\cos\hat{\alpha} = 1$, and

$$\hat{\mathbf{a}}^2 = \hat{\mathbf{b}}^2 = -1 \tag{71}$$

Reversing the order of products in Eq.(70) changes the sense of the dual angle $\hat{\alpha}$ and we can write

$$\hat{\mathbf{b}}\hat{\mathbf{a}} = -(\cos\hat{\alpha} + \hat{\mathbf{e}}\sin\hat{\alpha}) \tag{72}$$

Finally, postmultiplying both sides of the above equation with the unit line vector $\hat{\mathbf{a}}$, and using the property of Eq.(71), we obtain the equation

$$\hat{\mathbf{b}} = (\cos\hat{\alpha} + \hat{\mathbf{e}}\sin\hat{\alpha})\hat{\mathbf{a}} \tag{73}$$

where $(\cos\hat{\alpha} + \hat{\mathbf{e}}\sin\hat{\alpha})$ is the *orthogonal dual unit quaternion screw displacement operator*, rotating/translating the unit dual vector $\hat{\mathbf{a}}$ into the unit line vector $\hat{\mathbf{b}}$ along the screw axis $\hat{\mathbf{e}}$, which is *perpendicular* to $\hat{\mathbf{a}}$ (and thus $\hat{\mathbf{b}}$). Note that the orthogonal operator operates on a dual vector from the left only. This is in contrast to the general dual unit quaternion screw operator, which acts on a dual vector as a similarity transformation (Section 3.2.2, Eq.(33)).

The fundamental idea behind being able to describe robot manipulator kinematics using only orthogonal screw displacements is the fact that for each of the two screws comprising a general link transformation operator $\hat{\mathbf{A}}_i$ between frames $\mathcal{F}_{i-1}$ and $\mathcal{F}_i$ (Section 5.1, Eq.(57)), the screw axis is in fact perpendicular to the unit line vector being transformed. This, of course, is due to the fact that we are essentially screw displacing one axis of an orthogonal coordinate frame along another. For instance, the joint displacement screw $\text{Screw}(\hat{\theta}_i)$ is an orthogonal screw displacement

of the dual axis $\hat{\mathbf{x}}_{i-1}$ into $\hat{\mathbf{x}}_i$ along $\hat{\mathbf{z}}_{i-1}$. Similarly, the twist screw Screw($\hat{\alpha}_i$) is an orthogonal screw displacement of the dual axis $\hat{\mathbf{z}}_{i-1}$ into $\hat{\mathbf{z}}_i$ along $\hat{\mathbf{x}}_i$.

We can now describe the kinematic configuration of an n-d.o.f. manipulator as a pair of inter-dependent chains of orthogonal screw displacements of the dual axes $\hat{\mathbf{x}}$ and $\hat{\mathbf{z}}$. For a 6 d.o.f. manipulator, we obtain the following set of *transformation equations*

$$\begin{aligned}
\hat{\mathbf{x}}_1 &= \mathbf{Q}_{10} * \hat{\mathbf{x}}_0 = (\cos\hat{\theta}_1 + \hat{\mathbf{z}}_0 \sin\hat{\theta}_1) * \hat{\mathbf{x}}_0 \\
\hat{\mathbf{x}}_2 &= \mathbf{Q}_{21} * \hat{\mathbf{x}}_1 = (\cos\hat{\theta}_2 + \hat{\mathbf{z}}_1 \sin\hat{\theta}_2) * \hat{\mathbf{x}}_1 \\
\hat{\mathbf{x}}_3 &= \mathbf{Q}_{32} * \hat{\mathbf{x}}_2 = (\cos\hat{\theta}_3 + \hat{\mathbf{z}}_2 \sin\hat{\theta}_3) * \hat{\mathbf{x}}_2 \\
\hat{\mathbf{x}}_4 &= \mathbf{Q}_{43} * \hat{\mathbf{x}}_3 = (\cos\hat{\theta}_4 + \hat{\mathbf{z}}_3 \sin\hat{\theta}_4) * \hat{\mathbf{x}}_3 \\
\hat{\mathbf{x}}_5 &= \mathbf{Q}_{54} * \hat{\mathbf{x}}_4 = (\cos\hat{\theta}_5 + \hat{\mathbf{z}}_4 \sin\hat{\theta}_5) * \hat{\mathbf{x}}_4 \\
\hat{\mathbf{x}}_6 &= \mathbf{Q}_{65} * \hat{\mathbf{x}}_5 = (\cos\hat{\theta}_6 + \hat{\mathbf{z}}_5 \sin\hat{\theta}_6) * \hat{\mathbf{x}}_5
\end{aligned} \tag{74}$$

$$\begin{aligned}
\hat{\mathbf{z}}_1 &= \mathbf{Q}_1 * \hat{\mathbf{z}}_0 = (\cos\hat{\alpha}_1 + \hat{\mathbf{x}}_1 \sin\hat{\alpha}_1) * \hat{\mathbf{z}}_0 \\
\hat{\mathbf{z}}_2 &= \mathbf{Q}_2 * \hat{\mathbf{z}}_1 = (\cos\hat{\alpha}_2 + \hat{\mathbf{x}}_2 \sin\hat{\alpha}_2) * \hat{\mathbf{z}}_1 \\
\hat{\mathbf{z}}_3 &= \mathbf{Q}_3 * \hat{\mathbf{z}}_2 = (\cos\hat{\alpha}_3 + \hat{\mathbf{x}}_3 \sin\hat{\alpha}_3) * \hat{\mathbf{z}}_2 \\
\hat{\mathbf{z}}_4 &= \mathbf{Q}_4 * \hat{\mathbf{z}}_3 = (\cos\hat{\alpha}_4 + \hat{\mathbf{x}}_4 \sin\hat{\alpha}_4) * \hat{\mathbf{z}}_3 \\
\hat{\mathbf{z}}_5 &= \mathbf{Q}_5 * \hat{\mathbf{z}}_4 = (\cos\hat{\alpha}_5 + \hat{\mathbf{x}}_5 \sin\hat{\alpha}_5) * \hat{\mathbf{z}}_4 \\
\hat{\mathbf{z}}_6 &= \mathbf{Q}_6 * \hat{\mathbf{z}}_5 = (\cos\hat{\alpha}_6 + \hat{\mathbf{x}}_6 \sin\hat{\alpha}_6) * \hat{\mathbf{z}}_5
\end{aligned} \tag{75}$$

Note that the two sets of equations depend on each other, *i.e.*, in order to compute the new location of $\hat{\mathbf{z}}_i$ with respect to $\mathcal{F}_0$, we need to first obtain the description of the dual vector $\hat{\mathbf{x}}_i$. The newly obtained axis $\hat{\mathbf{z}}_i$, in turn, is used to compute $\hat{\mathbf{x}}_{i+1}$. This is consistent with the definition of the Denavit-Hartenberg parameters, where in moving from frame $\mathcal{F}_{i-1}$ to $\mathcal{F}_i$, the joint displacement screw (aligning $\hat{\mathbf{x}}_{i-1}$ and $\hat{\mathbf{x}}_i$) is performed before the twist screw (aligning $\hat{\mathbf{z}}_{i-1}$ and $\hat{\mathbf{z}}_i$).

Next, we choose the base link frame $\mathcal{F}_0$ as the fixed reference coordinate frame, represented by the base vectors $\mathbf{i}, \mathbf{j}, \mathbf{k}$. Then, we have

$$\begin{aligned}
\hat{\mathbf{x}}_0 &= \mathbf{i} \\
\hat{\mathbf{z}}_0 &= \mathbf{k}
\end{aligned} \tag{76}$$

$$(\hat{\mathbf{z}}_0\hat{\mathbf{x}}_0) \;=\; \mathbf{j}$$

and we can substitute $\mathbf{i},\mathbf{j},\mathbf{k}$ into Eqs.(74,75). Through successive substitutions we can now express $\hat{\mathbf{x}}_6$ and $\hat{\mathbf{z}}_6$ in terms of the base coordinate system $\mathcal{F}_0$. The resulting dual vector equations are functions of the six joint variables $\vartheta_1, \vartheta_2, \ldots, \vartheta_6$. In order to extract their values, we now set the two symbolic expressions for $\hat{\mathbf{x}}_6$ and $\hat{\mathbf{z}}_6$ equal to the known (*i.e.*, given) numeric values $\hat{\mathbf{T}}_{6_x}$ and $\hat{\mathbf{T}}_{6_z}$ of the two dual vectors. We thus obtain a pair of dual vector equations of the form

$$\begin{aligned} \hat{\mathbf{x}}_6 &= \mathbf{i}\hat{x}_{6_x} + \mathbf{j}\hat{x}_{6_y} + \mathbf{k}\hat{x}_{6_z} = \hat{\mathbf{T}}_{6_x} \\ \hat{\mathbf{z}}_6 &= \mathbf{i}\hat{z}_{6_x} + \mathbf{j}\hat{z}_{6_y} + \mathbf{k}\hat{z}_{6_z} = \hat{\mathbf{T}}_{6_z} \end{aligned} \tag{77}$$

where $\hat{\mathbf{T}}_{6_x}$ and $\hat{\mathbf{T}}_{6_z}$ are 6-tuples of numeric Plücker coordinates corresponding to the given location of the end-effector coordinate axes $\hat{\mathbf{x}}_6$ and $\hat{\mathbf{z}}_6$. Each row of Eq.(77) can be separated into three dual equations. However, due to mutual orthogonality of the base frame unit vectors $\mathbf{i},\mathbf{j},\mathbf{k}$, only two of the three equations are independent in each case. The two dual vector equations of Eq.(77) thus contribute two dual equations each, for a total of eight real equations. From this set of displacement equations we can then compute the values of the joint variables ϑ_i, $1 \leq i \leq 6$, completing the solution of the inverse kinematics problem.

Due to rather formidable complexity of the resulting expressions, I have not actually performed the complete analysis to see if the set of equations mentioned above is indeed sufficient. This would be an interesting future project, especially since, to the best of my knowledge, a purely orthogonal screw analysis of an open-loop manipulator chain has not been done.

6 Discussion and Conclusion

The purpose of this work was to survey the available literature in the area of line-oriented spatial screw transformations and perform an analysis of the proposed mathematical formalisms in terms of their computational behavior and storage requirements. Four distinct representations of a general spatial screw displacement were considered: (1) dual orthogonal 3×3 matrix, (2) dual unit quaternion, (3) dual special unitary 2×2 matrix, and (4) dual Pauli spin matrices. Both sequential and parallel algorithms for effecting a general screw displacement of a line in space, and

composing two general screw displacement operators, were presented and analyzed in terms of computational expense.

Referring to the tabular summary of the results in Section 4.4, we see that within the framework of sequential computing paradigm, the dual unit quaternion screw displacement operator provides the most compact and the most computationally efficient formalism for the two operations under consideration. Although the dual 3×3 matrix provides a very efficient means of screw displacing lines, it ranks only third (behind dual Pauli spin matrix operator) due to its comparatively high composition cost and representational redundancy.

For parallel applications, the choice of the optimal operator is less obvious. As in the case of sequential execution, the dual special unitary matrix operator is clearly the least desirable option. Similarly, the high storage cost of dual orthogonal matrices again eliminates them from serious consideration in comparison with the dual unit quaternion or dual Pauli spin matrix operators. However, the final choice between the latter two formalisms is quite arbitrary. The dual Pauli spin matrix operator is one cycle faster during composition of operators, but uses more processors for both of the operations. In the absence of a restriction on the number of independent processors, dual Pauli spin matrices thus represent the optimal representation, but the advantage over the dual unit quaternions is, from a practical standpoint, insignificant.

In terms of the application of screw displacements to robot kinematics, line oriented screw operators offer a natural and intuitive description of the spatial relationships between successive coordinate frames along the manipulator linkage. Their main appeal is that these operators operate on *lines* rather than *points* and are hence conceptually well suited for an application primarily concerned with rotations and translations of coordinate axes attached to the manipulator links. However, it is well known that point transformation methods suffice for kinematic analysis of arbitrary spatial linkages, and many such solutions have been produced ([Paul,1981],[Lee,1982],*etc.*). These point transformation based solutions tend to compare favorably with the corresponding solutions using dual quantities and screw displacements of lines. This is reflected both in the conciseness and elegance of derivations, as well as in the compactness and brevity of the results (*e.g.*, inverse kinematics joint variable equations). It is therefore my conclusion, based on my

exposure and experience with line-based screw displacements, that they are computationally and analytically (*i.e.*, for purposes of theoretical analyses and derivations) less advantageous than their point transformation counterparts.

References

[1] Roth B. Screws, motors, and wrenches that cannot be bought in a harware store. In Brady and Paul, editors, *Robotics Research*, pages 679–693, MIT Press, 1984.

[2] J. Denavit and R. S. Hartenberg. A kinematic notation for lower-pair mechanisms based on matrices. *Journal of Applied Mechanics*, 215–221, June 1955.

[3] Study E. *Geometrie der Dynamen.* Teubner, Leipzig, 1901.

[4] Lee C. S. G. Robot arm kinematics, dynamics, and control. *Computer, IEEE Trans.*, 62–80, Dec. 1982.

[5] Sir William R. Hamilton. *Elements of Quaternions.* Volume I & II, Chelsea Publishing Company, New York, NY, 1869.

[6] Berthold K. P. Horn. Closed-form solution of absolute orientation using unit quaternions. *Journal of the optical society of America*, 4, April 1987.

[7] Funda J. *Quaternions and Homogeneous Transforms in Robotics.* Master's thesis, University of Pennsylvania, 1988. MS-CIS-88-06.

[8] Rooney J. A comparison of representations of general spatial screw displacements. *Environment and planning B*, 5:45–88, 1978.

[9] Rooney J. On the principle of transference. In *Proceedings of the Fourth World Congress on the Theory of Machines and Mechanisms*, pages 1088–1092, 1975.

[10] Rooney J. On the three types of complex number and planar transformations. *Environment and planning B*, 5:89–99, 1978.

[11] Rooney J. A survey of representations of spatial rotation about a fixed point. *Environment and planning B*, 4:185–210, 1977.

[12] Clifford W. K. Preliminary scetch of biquaternions. In *Proceedings of the London Mathematical Society*, pages 381–395, 1873.

[13] Brand L. *Vector and Tensor Analysis.* John Wiley, New York, 1947.

[14] McCarthy J. M. Dual orthogonal matrices in manipulator kinematics. In J. M. McCarthy, editor, *Kinematics of Robot Manipulators*, pages 43–49, MIT Press, 1987.

[15] Yaglom I. M. *Complex numbers in Geometry.* Academic Press, New York, 1968.

[16] Kotelnikov A. P. Screw calculus and some applications to geometry and mechanics. Annals of the Imperial University of Kazan, 1895.

[17] Richard P. Paul. *Robot Manipulators: Mathematics, Programming and Control.* MIT Press, 1981.

[18] Pennock G. R. and Yang A. T. Application of dual-number matrices to the inverse kinematics problem of robot manipulators. *Journal of Mechanisms, Transmissions, and Automation in Design*, 107:201–208, 1985.

[19] Ball R.S. *A Treatise on the Theory of Screws.* Cambridge Univ. Press, 1900.

[20] Yang A. T. and Freudenstein F. Application of dual-number quaternion algebra to the analysis of spatial mechanisms. *Transactions of the ASME, Journal of Applied Mechanics*, 31:300–308, 1964.