A FUNDAMENTALLY NEW VIEW OF THE PERSPECTIVE THREE-POINT POSE PROBLEM

Michael Q. Rieck[1]
[1]*Mathematics and Computer Science Department, Drake University, Des Moines, IA 50310, USA*
mrieck@drake.edu

Keywords: P3P: perspective: pose: camera: tracking: resection: danger cylinder

Abstract: The Perspective Three-Point Pose Problem (P3P) is an old and basic problem in the area of camera tracking. While methods for solving it have been largely successful, they are subject to erratic behavior near the so-called "danger cylinder." Another difficulty with most of these methods is the need to select the physically correct solution from among various mathematical solutions. This article presents a new framework from which to study P3P for non-collinear control points, particularly near the danger cylinder. A multivariate Newton-Raphson method to approximately solve P3P is introduced. Using the new framework, this is then enhanced by adding special procedures for handling the problematic behaviour near the danger cylinder. It produces a point on the cylinder, a compromise between two nearly equal mathematical solutions, only one of which is the camera's actual position. The compromise diminishes the risk of accidentally converging to the other nearby solution. However, it does impose the need, upon receding from the danger cylinder vicinity, to make a selection between two possible approximate solution points. Traditional algebraic methods depend on correctly selecting from up to four points, each time the camera position is recomputed. In the new iterative method, selecting between just two points is only occasionally required. Simulations demonstrate that a considerable improvement results from using this revised method instead of the basic Newton-Raphson method.

1 INTRODUCTION

1.1 Overview of P3P

The Perspective Three-Point Pose Problem (P3P) is concerned with the determination of the position and orientation of a camera, based on the images of three known fixed points, called "control points." Attention in this article will be limited to the pinhole camera model. In the camera's reference frame, the three perspective lines connecting the camera's optical center (center of perspective projection) to the three control points are presumed to be known.

While camera tracking usually involves using more than just three control points, unfortunate circumstances can arise that limit the camera's view to only three points for extended periods of time. This limitation on control points presents a significant challenge to accurately following the camera in a computer system. This is especially true when reliable estimates need to be made rapidly due to continuous camera movement.

The difficult part of P3P is the determination of the camera's position, after which its orientation can readily be determined, as discussed in Subsection 2.1. Thus, this article will focus on the position issue. Using long established "algebraic methods," mathematical reasoning is capable of providing a small set of (up to four) possible camera positions.

However, correctly choosing from among these solutions has long been a thorny issue, and it is impossible using solely the P3P data. When only three control points are viewable, it is often still possible though to make an informed guess as to which position is the correct one, based on additional information such as known recent camera positions.

Less commonly considered iterative methods, such as the ones discussed in Section 5, avoid the need to select from among several options, but only provide approximate solutions. They can also break down and end up tracking the wrong solution. Nevertheless, the simpler nature of the computations involved, and the avoidance of a selection process would make them preferable to algebraic methods in certain computationally limited circumstances.

All current methods for solving P3P suffer instabilities due to multiple solutions and computational error when the camera's optical center is too close to the so-called "danger cylinder" region. The danger cylinder is the circular cylinder that contains the three control points, and whose axis is perpendicular to the plane containing these points.

Repeated solutions to the systems of P3P equations occur when the optical center is on the danger cylinder, resulting in singular behavior. Several studies of this phenomenon have been made, some of which are mentioned in the next subsection. The present article too is heavily focused on contending with the obstacles that result when the camera is too close to the danger cylinder.

1.2 Related Work

Since it was first introduced and solved (Grunert, 1841), various efforts have been made to better understand P3P and its underlying system of equations. Alternative methods for solving P3P have also been introduced, though these either proceeded along somewhat similar lines as the original solution (Haralick et al., 1994), or else employ some sort of approximation technique, such as (DeMenthon and Davis, 1992), (Fung and Wong, 1998) and Appendix A.3 of (Fischler and Bolles, 1981). The latter two are among the very few efforts involving iterative methods. Some other early work, much of it motivated by aerial reconnaissance concerns, can be found in (Merritt, 1949), (Müller, 1925), (Smith, 1965) and (Thompson, 1966).

Several studies have classified solutions, such as (Faugère et al., 2008), (Gao et al., 2003), (Sun and Wang, 2010), (Tang et al., 2008), (Tang and Liu, 2009), (Wolfe et al., 1991), (Zhang and Hu, 2005). Some of the more recent algorithms for solving P3P, and generalizations and restrictions of it, can be found in (Nistér, 2007), (Pisinger and Hanning, 2007), (Rieck, 2010), (Rieck, 2011), (Rieck, 2012), (Xiaoshan and Hangfei, 2001). A recent reexamination of the danger cylinder phenomenon can be found in (Zhang and Hu, 2006).

1.3 Layout of this Article

This article revisits some material that was previously presented in (Rieck, 2012). However, it does so using a far firmer foundation, and it provides considerably more detail, including proofs. The earlier paper was extremely brief, and just presenting certain facts, without proofs.

In particular, Section 3 (of the present work), on planar geometry, is entirely new, as is the whole approach to proving Theorem 1, in Section 4. Together this new material makes the theorem far more intuitive and accessible. Section 2 introduces this theorem, along with some foundational notation, concepts and relations.

Section 5, which is also new, introduces an iterative camera tracking technique based on P3P and Theorem 1. Specific details for this algorithm are found in the appendices. The results of simulations using this new method are also presented in Section 5.

2 ANALYSIS OF P3P

The camera model to be used throughout this article is the pinhole model, and the camera is presumed to be calibrated. The camera's image plane is presumed to have two perpendicular axes that pass through the image center. The focal length, that is, the distance between the optical center (center of perspective projection) and the image center, is a presumed to be known.

Discussions of camera tracking invariably entail multiple 3D-frames of reference. At any moment in time, moving between any two such coordinate systems is in principle simply a matter of applying the appropriate affine transformation. This is uniquely determined by specifying how any four points in general position are to be transformed.

We will begin with a fixed reference frame for the environment being viewed. The positions of the control points in this reference frame are presumed to be known.

2.1 Determining Orientation from Position

Suppose, for a moment, that the camera's position in the environmental reference frame can somehow be determined, though determining this is the principal problem of concern in this article. This means in particular that the distances from the control points to the optical center can readily be computed.

The environmental reference frame can then be translated so as to place the camera's optical center at the origin, without changing the directions of the axes. Let us refer to this as the "shifted" environmental reference frame. Now, initially, but probably incorrectly, suppose that in this reference frame, the camera is oriented so as to point along the x-axis's positive direction, with the image plane's first axis's positive direction matching the y-axis's positive direction, and the image plane's second axis's positive direction matching the z-axis's positive direction.

It is then possible to examine the actual images of the control points in the image plane, and together with the computed distances from the control points to the optical center, determine where the control points would need to be located so as to produce these images. That is, it is straightforward to figure out where the control points would need to be in the shifted environmental reference frame, based on the tentatively presumed camera pose.

Now, based also on a knowledge of where the control points actually are in the shifted environmental reference frame, versus where they should be when using the above orientation assumption, one can determine the unique rotation that moves one of these sets of points into the other set of points. This might be accomplished using Euler angles, but in any case, yields the camera's true orientation. This is why it was stated in the introduction that once the camera's position is determined, its orientation can also be readily determined.

2.2 Quantities and Systems

Let us begin a careful study of the essential P3P problem, focusing solely on the issue of finding the camera's position in the environmental frame of reference. When the three control points are not collinear, they lie on a unique circle, which is a basic fact from classical geometry. We will assume henceforth that the control points are not collinear. Several other studies, such as (Hanning et al., 2006), have focused instead on the special case where the control points are collinear.

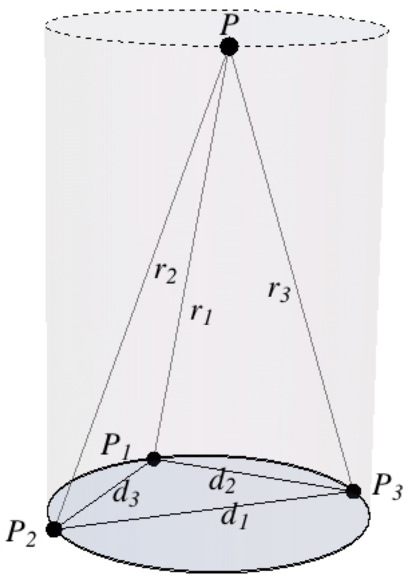

Figure 1: Danger cylinder (side view)

To simplify the notation, we may conveniently suppose that the unit of distance has been chosen so that this circle has radius one. The formulas to be presented in this article can easily be scaled so as to accommodate an arbitrary radius. In Theorem 1, just divide d_1, d_2, d_3, x, y and z by this radius.

For our purpose, it will be handy to fix a frame of reference for the environment, in which the three control points remain fixed. A Cartesian coordinate system will be set such that the three control points, P_1, P_2, P_3, lie on the unit circle centered about the origin, in the xy-plane. For $j = 1,2,3$, let

$$(x_j, y_j, 0) \;=\; (\cos\phi_j, \sin\phi_j, 0)$$

be the coordinates of P_j, with $-\pi \leq \phi_j \leq \pi$, and let

$$t_j \;=\; \tan(\phi_j/2)\,.$$

Also let d_j be the distance between the two control points other than P_j. From the standpoint of P3P, all these quantities are known a priori. Figure 1 shows the r_j and d_j, but in the special case where P is on the danger cylinder.

The unknown coordinates of the camera's optical center P will be denoted

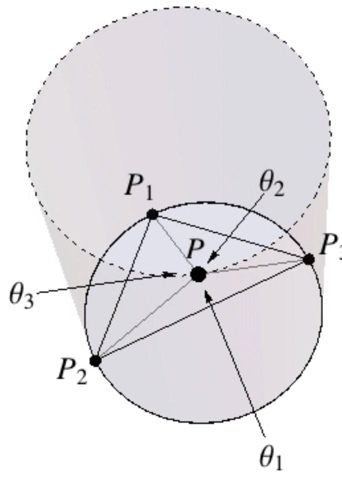

Figure 2: Danger cylinder (top-down view)

$$(x,y,z) \;=\; (\rho\cos\phi, \rho\sin\phi, z)\,.$$

with $\rho \geq 0$ and $-\pi \leq \phi \leq \pi$.

Let r_j be the distance between P and P_j ($j = 1,2,3$), which is also unknown. For $j = 1,2,3$, let θ_j be the angle at P created by the two rays to the two control points other than P_j. These angles are presumed to be known since they are easily computed from the camera images of the control points and the camera intrinsics. Figure 2 shows these angles, but in the special case when P is on the danger cylinder.

Let

$$c_j = \cos\theta_j \;\text{ and }\; s_j = \sin\theta_j \;\; (j = 1,2,3)\,.$$

An important geometric quantity that will be discussed later, and that can be computed based solely on the (known) cosines c_1, c_2 and c_3 is

$$\eta \;=\; \sqrt{1 - c_1^2 - c_2^2 - c_3^2 + 2c_1c_2c_3}\,.$$

With the setup described here, the danger cylinder is given by the equation $x^2 + y^2 = 1$. It is a well-studied fact (Zhang and Hu, 2006) that when the optical center is on or near the danger cylinder, traditional techniques for solving the P3P run into difficulties caused by imprecision in numerical computations. Figures 1 and 2 shows the situation when the optical center is on the danger cylinder, and above the horizontal plane containing the control points.

The problem of determining the camera's optical center P is classically broken down into a pair of systems of quadratic polynomials, as follows:

$$\begin{cases} r_2^2 + r_3^2 - 2c_1r_2r_3 \;=\; d_1^2 \\ r_3^2 + r_1^2 - 2c_2r_3r_1 \;=\; d_2^2 \\ r_1^2 + r_2^2 - 2c_3r_1r_2 \;=\; d_3^2 \end{cases} \tag{1}$$

$$\begin{cases} (x-x_1)^2 + (y-y_1)^2 + z^2 \;=\; r_1^2 \\ (x-x_2)^2 + (y-y_2)^2 + z^2 \;=\; r_2^2 \\ (x-x_3)^2 + (y-y_3)^2 + z^2 \;=\; r_3^2 \end{cases} \tag{2}$$

The first system just amounts to three applications of the Law of Cosines. The c_j and d_j are known quantities, and one must solve for the unknowns r_j. Unfortunately, this system is rather tedious to solve. What is worse is that in general there will be multiple real-valued solutions, and there can be as many as four possible values for each of r_1^2, r_2^2, and r_3^2. Of course only one of the solutions corresponds to the camera's actual position. Additional information would be needed to decide which of the mathematical solutions is the correct physical solution.

Solving the first system is traditionally handled by eliminating two of the r_j, say r_1 and r_2, and thus obtaining a quartic polynomial in r_3^2, or perhaps some closely related value (Haralick et al., 1994). One of the present article's referees has suggested a nice way to obtain a quartic in r_3^2. One can multiply each of the three equations in (1) by each of the six quantities 1, r_1, r_2, r_1^2, r_1r_2, r_2^2, thus obtaining eighteen equations. These can be viewed as homogeneous linear equations in the fifteen quantities 1, r_1, r_2, r_1^2, r_1r_2, r_2^2, r_1^3, $r_1^2r_2$, $r_1r_2^2$, r_2^3, r_1^4, $r_1^3r_2$, $r_1^2r_2^2$, $r_1r_2^3$, r_2^4, with coefficients involving c_1, c_2, c_3, d_1, d_2, d_3 and r_3. The 18-by-15 coefficient matrix of this homogeneous system must have vanishing 15-by-15 sub-determinants. This requirement amounts to the vanishing of a quartic polynomial in r_3^2 that does not involve r_1 or r_2.

Once the r_j are known, the second system (used also in many other circumstances, such as GPS navigation) can then readily be solved for the camera's position coordinates x, y and z. In fact, the third equation in (2) can be subtracted from each of the first two equations in (2) to obtain two linear equations in x and y. One can visualize the second system as dealing with the intersection of three specific spheres in space.

This approach, first solving (1) and then solving (2), works fairly well, as long as the control points are reasonably far apart, the optical center is reasonably close to the control points, and the optical center is reasonably far from the danger cylinder. The exact meaning of these conditions depends of course on the precision used in performing floating point computations. In practice, camera issues such as pixelation also causes imprecision that can adversely affect the results.

Since we are ultimately interested in obtaining x, y and z, with r_1, r_2 and r_3 only serving as intermediaries, one might consider eliminating the latter to obtain a single system of three equations in x, y and z. This is easily accomplished as follows. Begin by rewriting the equations in (1) by putting the terms involving the cosines on one side and the other terms on the other side. For instance, the first equation becomes $2c_1r_2r_3 = r_2^2 + r_3^2 - d_1^2$. Now square both sides of these equations, and replace each r_j^2 with the left side of the matching equation in (2). While this does eliminate the r_j, the resulting equations involve fourth degree polynomials in the unknowns x, y and z.

One would like to have a Gröbner or similar basis for the new system of equations, involving x, y and z as the only unknowns. However, direct methods for computing such a basis are rather tedious, and the author is unaware of any successful efforts along these lines. The next subsection provides a useful alternative approach, one which avoids finding a Gröbner basis, and which makes a very compelling connection with the danger cylinder.

2.3 The Principal Theorem

A radically different, and in some respects simpler and faster, approach to P3P, will be developed in this article. It is founded on the following "principal theorem." The theorem relates a simple rational combination of the known sines and cosines s_1, s_2, s_3, c_1, c_2 and c_3, and the known separation distances d_1, d_2 and d_3, to a two-part rational function of the desired unknowns x, y, z. It will be proved later in Section 4.

In order to smoothly state the theorem, it will be helpful to introduce a homogeneous quadratic polynomial of x and y that is parameterized by an arbitrary angle ω, as follows:

$$\Sigma(\omega;\, x,y) \;=\; (\sin\omega)(y^2 - x^2) + (\cos\omega)(2xy)\,.$$

Theorem 1.

$$\left[d_1^2 s_2^2 \;-\; d_2^2 s_1^2\right] \Big/ \eta^2 \quad = \quad A(\phi_1,\phi_2,\phi_3;x,y) \;+\; B(\phi_1,\phi_2,\phi_3;x,y)\;\frac{1-x^2-y^2}{z^2}\;, \qquad (3)$$

where

$$A(\phi_1,\phi_2,\phi_3;x,y) \;=\; \csc\left(\frac{\phi_1-\phi_2}{2}\right)\cdot \Sigma\left(\frac{\phi_1+\phi_2+2\phi_3}{2};\; x+x_3,\, y+y_3\right) \qquad (4)$$

and

$$B(\phi_1,\phi_2,\phi_3;x,y) \;=\; \frac{d_1^2-d_2^2}{4} \;-\; \csc\left(\frac{\phi_1-\phi_2}{2}\right)\cdot \Sigma\left(\frac{\phi_1+\phi_2+2\phi_3}{2};\; x \;-\; \frac{x_1+x_2}{2},\, y \;-\; \frac{y_1+y_2}{2}\right). \qquad (5)$$

The above remains true when the subscripts 1, 2 and 3 are permuted. This generally results in three distinct (though linearly dependent) equations.

Notice that the two expressions $A(\phi_1,\phi_2,\phi_3;x,y)$ and $B(\phi_1,\phi_2,\phi_3;x,y)$ are simply quadratic polynomials in x and y, with known coefficients. Notice too that on the danger cylinder $x^2+y^2=1$, equation (3) becomes simply $(d_1^2s_2^2 - d_2^2s_1^2)/\eta^2 = A(\phi_1,\phi_2,\phi_3;x,y)$. Also, the quantity $z^2/(1-x^2-y^2)$ can be replaced by another variable W, thus eliminating z, and resulting in three cubic equations in x, y and W that are linear in W and quadratic in x and y.

2.4 Repeated solutions

In (Rieck, 2011), system (1) is treated as the equations for transforming a coordinate system (r_1,r_2,r_3) to a coordinate system (c_1,c_2,c_3), and the determinant of the Jacobian matrix J (with entries $\partial c_i/\partial r_j$) is computed and shown to vanish on and only on the danger cylinder.

It is further stated, but not proved, there that repeated solutions to system (1), solving for (r_1,r_2,r_3) given (c_1,c_2,c_3), occur on and only on the danger cylinder. This is proved here, as follows. Note though that (Thompson, 1966) and (Zhang and Hu, 2006) already provide a geometric sense of the significance of danger cylinder and its connection with J.

Given a point (x,y,z), let (r_1,r_2,r_3) be as in (2). Let (c_1,c_2,c_3) then be as in (1). Consider some (u_1,u_2,u_3) that also satisfies (1) when u_j stands in place of r_j ($j = 1,2,3$). If we express each c_i in terms of r_1,r_2,r_3, and also in terms of u_1,u_2,u_3, we can eliminate the c_i to obtain three equations relating r_1,r_2,r_3,u_1,u_2 and u_3, each of which can be put in the form of a polynomial equation (by multiplying by denominators).

Let $p_i = p_i(r_1,r_2,r_3,u_1,u_2,u_3) = 0$, for $i = 1,2,3$, be these three polynomial equations, scaled so that the coefficient of r_2^2 (and also r_3^2) in p_1 is $-u_2u_3$, and so forth. Let M be the 3×3 matrix formed by taking the partial derivatives of the p_i with respect to u_j, and then setting each u_j to r_j ($i, j = 1,2,3$).

Proposition 1. *The matrix M can be obtained from the matrix J by multiplying the top row by $2r_2^2r_3^2$, multiplying the middle row by $2r_3^2r_1^2$, and multiplying the bottom row by $2r_1^2r_2^2$. Assuming that no r_j equals zero,* $\det(M) = 0$ *if and only* $\det(J) = 0$ *if and only if $x^2+y^2 = 1$. Hence (r_1,r_2,r_3) is a repeated solution to system (1) if and only if $x^2+y^2 = 1$.*

Proof. Proving the claimed connection between that matrix M and the matrix J just amounts to carrying out the described computations, and is straightforward. The fact that $\det(J) = 0$ if and only if $x^2+y^2 = 1$ was established in (Rieck, 2011). The vector (p_1, p_2, p_3) as a function of (u_1,u_2,u_3), with (r_1,r_2,r_3) fixed, vanishes at $(u_1,u_2,u_3) = (r_1,r_2,r_3)$. However, it is locally invertible there unless $\det(M) = 0$. The vector equation $(p_1, p_2, p_3) = (0,0,0)$ has a repeated solution at $(u_1,u_2,u_3) = (r_1,r_2,r_3)$ if and only if it is not locally invertible here, that is, if and only if $\det(M) = 0$, that is, if and only if $\det(J) = 0$, that is, if and only if $x^2+y^2 = 1$.

□

2.5 Preliminary Analysis

To analyze the setup more completely, a number of basic identities need to be established first. These follow immediately from standard trigonometric identities. They are compiled in the following lemma. This is followed by an important but immediate corollary.

Lemma 1. *The following identities hold:*

(a) $x_j = \cos\phi_j = (1-t_j^2)/(1+t_j^2) \quad (j = 1,2,3)$

(b) $y_j = \sin\phi_j = 2t_j/(1+t_j^2) \quad (j = 1,2,3)$

(c) $\cos(\phi_j/2) = (1+t_j^2)^{-1/2} \quad (j = 1,2,3)$

(d) $\sin(\phi_j/2) = t_j(1+t_j^2)^{-1/2} \quad (j = 1,2,3)$

(e)
$$\begin{aligned} d_1 &= [(x_2-x_3)^2+(y_2-y_3)^2]^{1/2} \\ &= 2\sin(\,|\phi_2-\phi_3|/2\,) \\ &= 2\,|t_2-t_3|\,[(1+t_2^2)(1+t_3^2)]^{-1/2} \\ d_2 &= [(x_3-x_1)^2+(y_3-y_1)^2]^{1/2} \\ &= 2\sin(\,|\phi_3-\phi_1|/2\,) \\ &= 2\,|t_3-t_1|\,[(1+t_3^2)(1+t_1^2)]^{-1/2} \\ d_3 &= [(x_1-x_2)^2+(y_1-y_2)^2]^{1/2} \\ &= 2\sin(\,|\phi_1-\phi_2|/2\,) \\ &= 2\,|t_1-t_2|\,[(1+t_1^2)(1+t_2^2)]^{-1/2} \end{aligned}$$

(f)
$$\begin{aligned} r_j &= [\,(x-x_j)^2+(y-y_j)^2+z^2\,]^{1/2} \\ &= [\,1+x^2+y^2+z^2 \\ &\quad -2xx_j-2yy_j\,]^{1/2} \\ &= \{\,1+x^2+y^2+z^2 \\ &\quad -[2(1-t_j^2)x+4t_jy]\,/\,(1+t_j^2)\,\}^{1/2} \\ &\qquad (j = 1,2,3) \end{aligned}$$

(g)
$$\begin{aligned} c_1 &= \cos\theta_1 = (r_2^2+r_3^2-d_1^2)\,/\,2r_2r_3 \\ c_2 &= \cos\theta_2 = (r_3^2+r_1^2-d_2^2)\,/\,2r_3r_1 \\ c_3 &= \cos\theta_3 = (r_1^2+r_2^2-d_3^2)\,/\,2r_1r_2 \end{aligned}$$

Proof. Item (a) can be established like so: $(1-t_j^2)/(1+t_j^2) = (1-\tan^2 {}^{\phi_j}\!/_2)\,/\,(1+\tan^2 {}^{\phi_j}\!/_2)$ $= (\cos^2 {}^{\phi_j}\!/_2 - \sin^2 {}^{\phi_j}\!/_2)\,/\,(\cos^2 {}^{\phi_j}\!/_2 + \sin^2 {}^{\phi_j}\!/_2)$ $= (\cos^2 {}^{\phi_j}\!/_2 - \sin^2 {}^{\phi_j}\!/_2) = \cos\phi_j$. Similarly for item (b): $2t_j/(1+t_j^2) = 2\tan {}^{\phi_j}\!/_2\,/\,(1+\tan^2 {}^{\phi_j}\!/_2) = 2\sin {}^{\phi_j}\!/_2 \cos {}^{\phi_j}\!/_2 = \sin\phi_j$. For item (c): $1+t_j^2 = 1+\tan^2 {}^{\phi_j}\!/_2 = \sec^2 {}^{\phi_j}\!/_2$. Because of the chosen range for ϕ_j, (c) now follows since $\cos {}^{\phi_j}\!/_2$ is nonnegative. Item (d) is now immediate since $\sin {}^{\phi_j}\!/_2 = \tan {}^{\phi_j}\!/_2 \cos {}^{\phi_j}\!/_2$. For (e), notice that the isosceles triangle whose vertices are P_2, P_3 and the origin, can be cut into two congruent right triangle, leading immediately to the fact that $d_1 = 2\sin(\,|\phi_2-\phi_3|/2\,)$. Using the basic trigonometry together with the preceding formulas, the complete claim concerning d_1 follows. Ditto for d_2 and d_3. Concerning (f), apply the distance formula and use the earlier formulas. Item (g) simply restates the Law of Cosines seen earlier.

□

Corollary 1. *The quantities x_1, x_2, x_3, y_1, y_2, y_3, r_1^2, r_2^2, r_3^2, d_1^2, d_2^2, d_3^2, c_1^2, c_2^2, c_3^2, s_1^2, s_2^2, s_3^2, $c_1c_2c_3$, $d_1d_2d_3$ and η^2 can all be expressed as rational functions of t_1, t_2, t_3, x, y and z.*

Proof. Each of these quantities can be immediately checked using the formulas in Lemma 1 and the definition of η.

□

Now, for P3P, it is supposed that the quantities $\phi_1, \phi_2, \phi_3, t_1, t_2, t_3, x_1, x_2, x_3, y_1, y_2, y_3, \theta_1, \theta_2, \theta_3, c_1, c_2, c_3, s_1, s_2, s_3, d_1, d_2, d_3$ and η are known, and that the goal is to determine the unknown optical center coordinates x, y and z. We are also assuming that ρ, ϕ, r_1, r_2 and r_3 are unknown. Henceforth, we will suppose that the control points and the optical center are not coplanar, so that $\eta > 0$ and $z \neq 0$.

Lemma 2. *$r_1 r_2 r_3 \eta = d_1 d_2 d_3 |z|/2$, which equals six times the volume of the tetrahedron whose vertices are the optical center and the three control points. This also equals the volume of the parallelepiped having these four points among its vertices, with each control point adjacent to the optical center along an edge of the parallelepiped.*

Proof. The volume of the parallelepiped equals the absolute value of the (scalar) triple product of the three vectors from the optical center to the control points. This can be expressed as follows:

$$\left| \det \begin{pmatrix} x_1 - x & y_1 - y & -z \\ x_2 - x & y_2 - y & -z \\ x_3 - x & y_3 - y & -z \end{pmatrix} \right| .$$

To see that $r_1 r_2 r_3 \eta$ is also this volume, divide each of the vectors by its length (r_j). This yields three normal vectors, $\hat{n}_1$, $\hat{n}_2$, $\hat{n}_3$. The Gramian of these vectors is the determinant

$$\det \begin{pmatrix} 1 & c_3 & c_2 \\ c_3 & 1 & c_1 \\ c_2 & c_1 & 1 \end{pmatrix} = \eta^2 ,$$

whose entries are the dot products $\hat{n}_i \cdot \hat{n}_j$. But this Gramian is well-known to equal the square of the volume of the parallelepiped defined by the three normal vectors. So this volume equals η. We must scale this by multiplying by $r_1 r_2 r_3$ to obtain the volume of the original parallelepiped.

Now, the tetrahedron's volume is of course one-third its height $|z|$ times the area of its base triangle, whose vertices are P_1, P_2 and P_3. This triangle has circumradius one, and hence, by a well-known formula, its area is $d_1 d_2 d_3/4$. The tetrahedron's volume is well-known to be one-sixth the volume of the corresponding parallelepiped. From this, we can obtain $d_1 d_2 d_3 \, |z| \, / \, 2$ as the volume of the parallelepiped.

□

2.6 The Quadratic Polynomial $\Sigma(\omega;\, x, y)$

Theorem 1 involves the homogeneous quadratic polynomial $\Sigma(\omega; x, y)$. In fact, it occurs in both (4) and (5). It will be helpful to introduce alternative ways of parameterizing this. For fixed parameters t_1, t_2, t_3, define the following:

$$\Sigma(t_1, t_2, t_3;\, x, y) \;=\; \tag{6}$$

$$\Big[\, (t_1 + t_2 + 2t_3 - 2t_1 t_2 t_3 - t_1 t_3^2 - t_2 t_3^2)\, (y^2 - x^2) \\ +2\,(1 - t_1 t_2 - 2t_1 t_3 - 2t_2 t_3 - t_3^2 + t_1 t_2 t_3^2)\, xy \,\Big] \\ \Big/ \; \Big[(1 + t_1^2)^{1/2} (1 + t_2^2)^{1/2} (1 + t_3^2) \Big] ,$$

$$\text{and} \quad \Sigma(t_1, t_2;\, x, y) \;=\; \Sigma(t_1, t_2, 0;\, x, y) \;=\; \tag{7}$$

$$\big[\, (t_1 + t_2)\, (y^2 - x^2) + 2\,(1 - t_1 t_2)\, xy \,\big] \\ \Big/ \; \Big[(1 + t_1^2)^{1/2} (1 + t_2^2)^{1/2} \Big] .$$

Lemma 3. *The following facts hold:*

- ⋄ $\Sigma(\omega;\, x, y) \;=\; (\sin\omega)(y^2 - x^2) + (\cos\omega)(2xy)$
  $= (\sin\omega)\,[-\rho^2 \cos(2\phi)] + (\cos\omega)\,[\rho^2 \sin(2\phi)]$
  $= \rho^2 \sin(2\phi - \omega)$
- ⋄ $\Sigma(\omega;\, x, y) \;=\; [(\cos\omega + 1)x + (\sin\omega)y] \cdot$
  $[(-\sin\omega)x + (\cos\omega + 1)y] \;/\; (\cos\omega + 1)$
  $= [(\cos\omega - 1)x + (\sin\omega)y] \cdot$
  $[(-\sin\omega)x + (\cos\omega - 1)y] \;/\; (\cos\omega - 1)$
- ⋄ $\Sigma((\phi_1 + \phi_2)/2;\, x, y) \;=\; \Sigma(t_1, t_2;\, x, y)$
- ⋄ $\Sigma((\phi_1 + \phi_2 + 2\phi_3)/2;\, x, y) \;=\; \Sigma(t_1, t_2, t_3;\, x, y)$

Proof. The formulas in the first item follow quickly from the double angle and sum-of-angle formulas for the sine and cosine functions. Recall here that $(x, y) = (\rho\cos\phi, \rho\sin\phi)$. For the second item, the factorizations of $\Sigma(\omega; x, y)$ into products of linear parts are straightforward to check, using just the Pythagorean trigonometric identity $\cos^2\omega + \sin^2\omega = 1$.

For the third item, let $\Delta = [(1 + t_1^2)(1 + t_2^2)]^{1/2}$. Now, from Lemma 1, we see that $\sin\omega =$

$\sin(\phi_1/2)\cos(\phi_2/2) + \cos(\phi_1/2)\sin(\phi_2/2) = (t_1 + t_2)/\Delta$, and also $\cos\omega = \cos(\phi_1/2)\cos(\phi_2/2) - \sin(\phi_1/2)\sin(\phi_2/2) = (1 - t_1 t_2)/\Delta$. This immediately yields $\Sigma(\omega; x,y) = \Sigma(t_1,t_2; x,y)$.

To establish the last item, substitute $\phi_1 + \phi_3$ and $\phi_2 + \phi_3$ for ϕ_1 and ϕ_2 in the third part. This amounts to substituting $(t_1+t_3)/(1-t_1t_3)$ and $(t_2+t_3)/(1-t_2t_3)$ for t_1 and t_2. The claim is then straightforward to deduce.

□

Corollary 2. *The polynomial* $\Sigma(\omega;\ x,y)$ *vanishes when* (x,y) *is a point on a line through the origin with slope either* $\tan(\omega/2)$ *or* $-\cot(\omega/2)$. *It does not vanish at any other points.*

Proof. $\Sigma(0;\ x,y) = 2xy$, which vanishes when and only when $x = 0$ or $y = 0$. More generally, we see from the formula $\Sigma(\omega;\ x,y) = \rho^2 \sin[2(\phi - \omega/2)]$ that $\Sigma(\omega;\ x,y)$ vanishes when and only when (x,y) is a point on a line through the origin with either slope $\tan(\omega/2)$ or slope $\tan(\omega/2 + \pi/2) = -\cot(\omega/2)$.

□

3 PLANE GEOMETRY FACTS

In order to help make sense of Theorem 1, and ultimately prove it, this section will explore a few results from Euclidean geometry. Later, in the next section, these results will be used in a three-dimensional setting, and combined with some solid geometry results, concluding with a short proof of Theorem 1.

3.1 Significant Points and Lines

Throughout this section, P_1, P_2 and P_3 continue to be fixed points on the unit circle, in the Cartesian plane (xy-plane), while P' will denote an arbitrary point in this plane. To make the connection with the P3P discussion, P' represents the orthogonal projection of the optical center P onto the xy-plane. The notation of Section 2 will be continued here.

Certain lines, based on P_1, P_2 and P_3, are of particular importance. The first two, denoted ℓ_1 and ℓ_2, pass through the origin, are perpendicular to each other, and one of them has slope $\tan[(\phi_1 + \phi_2 + 2\phi_3)/4]$. Note that if multiples of 2π are added to ϕ_1, ϕ_2 and ϕ_3, then $(\phi_1 + \phi_2 + 2\phi_3)/4$ changes by the addition of

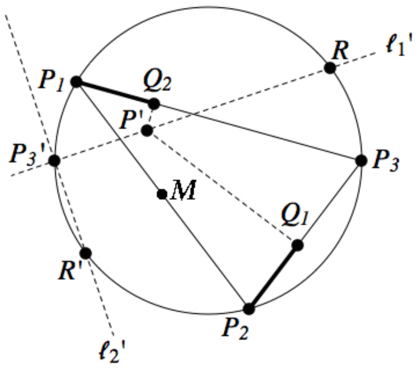

Figure 3: Lemma 4 Setup

a multiple of $\pi/2$, which might interchange ℓ_1 and ℓ_2. Thus $\{\ell_1, \ell_2\}$, as an unordered pair of lines, is well defined based solely on the points P_1, P_2 and P_3.

Let P_3' denote the point on the unit circle that is antipodal to P_3. Take ℓ_1' and ℓ_2' to be the lines parallel to ℓ_1 and ℓ_2, obtained by the translation that moves the origin to P_3'. Similarly, take ℓ_1'' and ℓ_2'' to be the lines parallel to ℓ_1 and ℓ_2, obtained by the translation that moves the origin to the midpoint M between P_1 and P_2. (Alternatively, the lines ℓ_1' and ℓ_2' could be defined as the lines that contain P_3' and one of the two points R and R' on the unit circle equidistant from P_1 and P_2, as seen in the proof of Lemma 4.)

3.2 Some Geometric Lemmas

Lemma 4. *If* P' *is on* $\ell_1' \cup \ell_2'$, *then the distance from* P_1 *to the orthogonal projection of* P' *onto the line* $\overleftrightarrow{P_1P_3}$ *equals the distance from* P_2 *to the orthogonal projection of* P' *onto the line* $\overleftrightarrow{P_2P_3}$.

Proof. Without loss of generality, by rotating the configuration as needed, we may assume that $\phi_3 = 0$. That is, we may assume that $P_3 = (1,0)$ and $P_3' = (-1,0)$.

Let Q_1 and Q_2 be the orthogonal projections of P' onto $\overleftrightarrow{P_2P_3}$ and $\overleftrightarrow{P_1P_3}$, respectively, as seen in Figure 3. In general, besides P_3', the unit circle intersects $\ell_1' \cup \ell_2'$ in two other points, which are labeled R and R' in the figure. (This is so except in the special case where either ℓ_1 or ℓ_2 is vertical, which means $\phi_1 + \phi_2 = 0$, and which can be treated as a limiting case.)

If $P' = P_3'$, then $Q_1 = P_2$ and $Q_2 = P_1$, since $\angle\, P_3'P_1P_3$ and $\angle\, P_3'P_2P_3$ are both right angles, by

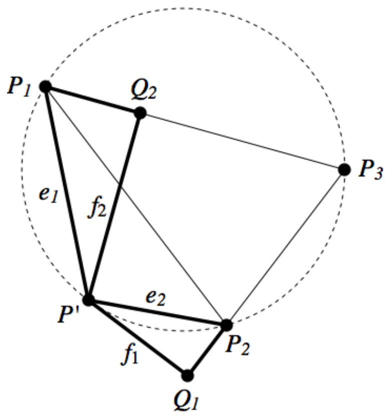

Figure 4: Lemma 5 Setup

the Inscribed Angle Theorem. Thus the claim is correct when $P' = P_3{}'$, since both distances are zero in this case.

The points R and R' are actually antipodal on the unit circle, and the slope of the line containing them (and the origin) is $\tan[(\phi_1+\phi_2)/2]$. To see this, again use the Inscribed Angle Theorem, comparing the inscribed angle $\angle RP_3{}'P_3$, which is $(\phi_1+\phi_2)/4 \bmod \pi/2$, and the central angle $\angle ROP_3$ (where O is the origin), which is therefore $(\phi_1+\phi_2)/2 \bmod \pi$. Ditto with R' in place of R. So R and R' are thus both equidistant from P_1 and P_2, and lie on $\ell_1{}' \cup \ell_2{}'$.

If $P' = R$, then by the Inscribed Angle Theorem again, the angles $\angle P_3P_1P'$ and $\angle P_3P_2P'$ are either similar or supplementary. If, R and P_3 occur on the same side of the line $\overleftrightarrow{P_1P_2}$, then the angles are similar (as in the figure); otherwise, they are supplementary (as in the figure with R' used instead of R).

Either way, checking each case, it is straightforward to see that the interior angles $\angle Q_2P_1P'$ and $\angle Q_1P_2P'$ of the right triangles $\triangle Q_2P_1P'$ and $\triangle Q_1P_2P'$ must be similar. The two right triangles are therefore similar. In fact, they are congruent since their hypotenuses have the same length, since P' is equidistant from P_1 and P_2. Therefore, the claim is true when $P' = R$, and likewise when $P' = R'$.

Now, when P' is a point between $P_3{}'$ and R (on a line), then Q_2 will be a point between P_1 and S_2, where S_2 is the orthogonal projection of R onto the line $\overleftrightarrow{P_1P_3}$. Clearly we have the following proportionality between lengths of segments: $\overline{P_3{}'P'}$ / $\overline{P_3{}'R}$ = $\overline{P_1Q_2}$ / $\overline{P_1S_2}$. Similarly, $\overline{P_3{}'P'}$ / $\overline{P_3{}'R}$ = $\overline{P_2Q_1}$ / $\overline{P_2S_1}$, where S_1 is the orthogonal projection of R onto the line $\overleftrightarrow{P_2P_3}$. But since $\overline{P_1S_2}$ = $\overline{P_2S_1}$, we see that $\overline{P_1Q_2}$ = $\overline{P_2Q_1}$. Thus the claim holds when P' is a point between $P_3{}'$ and R. This reasoning can be extended to cover all situations where P', $P_3{}'$ and R are collinear, and likewise where P', $P_3{}'$ and R' are collinear.

□

Lemma 5. *If P' is on the unit circle, then the product of the distance from P' to P_1 times the distance from P' to $\overleftrightarrow{P_2P_3}$ equals the product of the distance from P' to P_2 times the distance from P' to $\overleftrightarrow{P_1P_3}$.*

Proof. Here we will continue to use the notation in the proof of the previous lemma, but will also use the notation in Figure 4. Thus e_j and f_j denote the distance from P' to P_j and the distance from P' to Q_j, respectively ($j = 1,2$). We need to show that $e_1f_1 = e_2f_2$ when P' is on the unit circle.

Assuming that it is on the unit circle, the goal will be achieved if $\triangle Q_2P_1P'$ and $\triangle Q_1P_2P'$ can be shown to be similar right triangles. This is because the proportions $e_1 : f_2$ and $e_2 : f_1$ would then be the same. In the proof of Lemma 4, we showed that these right triangle were congruent when $P' = R$ (or R'). However, it was first established there that they were similar right triangle, by reasoning that did not require P' to be R or R', simply relying on the fact the P' was on the unit circle. This applies here as well.

□

We will continue to use the notation used in the proofs of the previous two lemmas, and in particular e_j and f_j ($j = 1,2$). Additionally, a quantity G is introduced "visually" as follows.

If P' is inside the unit circle, let G denote the square of the half length of the chord for the unit circle that has P' as midpoint. But if P' is outside the unit circle, consider a line that passes through P' and is tangent to the unit circle. Here let G be the negative of the squared distance along this line from P' to the point of intersection with the circle. Finally, if P' is on the unit circle, then just set $G = 0$. Now, it is easy to check that $G = 1 - x^2 - y^2$ in all these cases.

Lemma 6.

$$\diamond\ e_j = \left[(x-x_j)^2 + (y-y_j)^2\right]^{1/2}$$
$$\diamond\ f_1 = |(y_2-y_3)x + (x_3-x_2)y + x_2y_3 - x_3y_2| \,/\, d_1$$

(Similarly for f_2^2 and f_3^2)

Proof. The first item is immediate. To prove the second item, consider the triangle with vertices P', P_2 and P_3. Its area is $d_1 f_1/2$. Now, lift each of these three points one unit in the z-direction to obtain the points $(x,y,1)$, $(x_2,y_2,1)$ and $(x_3,y_3,1)$. The tetrahedron with these points and the origin as vertices has volume $d_1 f_1/6$. But reasoning as in the proof of Lemma 2, its volume also equals

$$\frac{1}{6}\left|\det\begin{pmatrix} x & y & 1\\ x_2 & y_2 & 1\\ x_3 & y_3 & 1\end{pmatrix}\right| .$$

□

Lemma 7.

$$e_2^2 - f_1^2 - e_1^2 + f_2^2 =$$

$$\sin\left(\frac{\phi_1-\phi_2}{2}\right)\Sigma(\psi;\, x+x_3, y+y_3),$$

where $\psi = (\phi_1+\phi_2+2\phi_3)/2$.

Proof. By Lemma 4, $(e_2^2 - f_1^2) - (e_1^2 - f_2^2)$ vanishes on $\ell_1{}' \cup \ell_2{}'$, and it is also a quadratic polynomial in x and y. But by Corollary 2, the same may be said of $\Sigma(\psi;\, x+x_3, y+y_3)$. Therefore, they both factor into linear polynomials, and in fact they are equal up to a constant factor.

The claim being made in the lemma is symmetric with respect to simultaneous (and equal) rotations of the control points and P'. That is, if each ϕ_j is replaced with $\phi_j + \alpha$, and ϕ is likewise replaced with $\phi+\alpha$, for a constant α, then both sides of the formula remain unaffected. So without loss of generality, we may assume that $\phi_2 = -\phi_1$, $x_2 = x_1$ and $y_2 = -y_1$.

When also $(x,y) = (x_3,y_3)$, we see that $(e_2^2 - f_1^2) - (e_1^2 - f_2^2) = e_2^2 - e_1^2 = 4y_1y_3$. But in this case, $\Sigma(\phi_3;\, x+x_3, y+y_3) = y_3[(2y_3)^2 - (2x_3)^2] + 2x_3(2x_3)(2y_3) = 4y_3(y_3^2+x_3^2) = 4y_3$. Therefore, for arbitrary (x,y), we get $(e_2^2 - f_1^2) - (e_1^2 - f_2^2) = y_1\,\Sigma(\phi_3;\, x+x_3, y+y_3)$. But $y_1 = \sin[(\phi_1-\phi_2)/2]$. So the claim in the lemma is true in the special setup. Since it is invariant under the rotations, it is also true in a general setup.

□

Lemma 8. *If P' is on $\ell_1{}''$ or on $\ell_2{}''$, then $16(e_1^2f_1^2 - e_2^2f_2^2) = (d_2^2 - d_1^2)d_3^2 G$. More generally, for any point P' in the plane,*

$$\frac{4(e_1^2f_1^2 - e_2^2f_2^2)}{d_3^2\,G} + \frac{d_1^2-d_2^2}{4} =$$

$$\csc\left(\frac{\phi_1-\phi_2}{2}\right)\Sigma\left(\psi;\, x-\frac{x_1+x_2}{2}, y-\frac{y_1+y_2}{2}\right),$$

where $\psi = (\phi_1+\phi_2+2\phi_3)/2$.

Proof. As polynomials in x and y, $e_1^2f_1^2 - e_2^2f_2^2$ has degree 4, while G has degree 2. Moreover, G must be a factor of $e_1^2f_1^2 - e_2^2f_2^2$ since the latter vanishes whenever G does (Lemma 5). Therefore $(e_1^2f_1^2 - e_2^2f_2^2)/G$ is a quadratic polynomial in x and y.

Now, as with the claim in Lemma 7, the claim in Lemma 8 is symmetric with respect to simultaneous (and equal) rotations of the control points and P'. Therefore, we may again assume that $\phi_2 = -\phi_1$, $x_2 = x_1$ and $y_2 = -y_1$. Then, by Lemma 6,

$$(e_1^2f_1^2 - e_2^2f_2^2)/G_1 =$$

$$\frac{y_1(x_1-x_3)^2[2x_3(x-x_1)y + y_3(y^2-x^2+2x_1x-1)]}{(1-x_1x_3+y_1y_3)(1-x_1x_3-y_1y_3)}$$

$$= y_1[2x_3(x-x_1)y + y_3(y^2-x^2+2x_1x-1)],$$

where the step is follows because $(1-x_1x_3+y_1y_3)(1-x_1x_3-y_1y_3) = (x_1-x_3)^2$, since $x_1^2+y_1^2 = 1$ and $x_3^2+y_3^2 = 1$. It is now straightforward to check that

$$\frac{4(e_1^2f_1^2 - e_2^2f_2^2)}{d_3^2\,G} + \frac{d_1^2-d_2^2}{4} =$$

$$\frac{e_1^2f_1^2 - e_2^2f_2^2}{y_1^2\,G} + y_1y_3 =$$

$$\frac{(x_1-x_3)^2\left(y_3[y^2-(x-x_1)^2] + 2x_3(x-x_1)y\right)}{y_1(1-x_1x_3+y_1y_3)(1-x_1x_3-y_1y_3)} =$$

$$\frac{(x_1-x_3)^2\,\Sigma(\phi_3;\, x-x_1, y)}{y_1(1-x_1x_3+y_1y_3)(1-x_1x_3-y_1y_3)} =$$

$$\frac{\Sigma(\phi_3;\, x-x_1, y)}{y_1} =$$

$$\csc\left(\frac{\phi_1-\phi_2}{2}\right)\Sigma\left(\psi;\, x-\frac{x_1+x_2}{2}, y-\frac{y_1+y_2}{2}\right).$$

The lemma is true for the special setup here. Thus it is also true in general.

□

4 SOLID GEOMETRY FACTS

The notation for the P3P setup, from the previous sections, will continue to be used here, and it will be assumed that P' is the projection of P onto the xy-plane. For $i, j = 1,2,3$ with $i \neq j$, we will let c_{ij} and s_{ij} denote the cosine and sine of the angle θ_{ij} at P_i, between the ray directed towards P and the ray directed towards P_k, where the set $\{i, j, k\}$ equals the set $\{1,2,3\}$. Here now are a number of relationships of importance in establishing Theorem 1.

4.1 More Needed Relationships

Lemma 9. *For $i, j = 1,2,3$ with $i \neq j$, the following relations hold:*

- ◇ $s_i^2 + c_i^2 = 1$
- ◇ $s_{ij}^2 + c_{ij}^2 = 1$
- ◇ $e_j^2 + z^2 = r_j^2$
- ◇ $e_i^2 - f_j^2 = r_i^2 c_{ij}^2$
- ◇ $(d_1^2 s_2^2 - d_2^2 s_1^2) \,/\, \eta^2 = 4 r_1^2 r_2^2 (c_{21}^2 - c_{12}^2) \,/\, d_3^2 z^2$

Proof. The first three items are straightforward to check, and all involve measuring the sides of right triangles, and applying the Pythagoras Theorem. For the fourth, consider the right triangles $\triangle P_i Q_j P$ and $\triangle P_i Q_j P'$. The last item can be established through a series of manipulations, involving algebra, basic trigonometry, and Lemma 2. Here are the steps:

$$\begin{aligned}
&(d_1^2 s_2^2 - d_2^2 s_1^2) \,/\, \eta^2 = \\
&r_1^2 r_2^2 (d_1^2 s_2^2 - d_2^2 s_1^2) \,/\, r_1^2 r_2^2 \eta^2 = \\
&[r_2^2 d_1^2 (r_1^2 s_2^2) - r_1^2 d_2^2 (r_2^2 s_1^2)] \,/\, r_1^2 r_2^2 \eta^2 = \\
&\qquad\qquad \text{(by Law of Sines)} \\
&[r_2^2 d_1^2 (d_2^2 s_{32}^2) - r_1^2 d_2^2 (d_1^2 s_{31}^2)] \,/\, r_1^2 r_2^2 \eta^2 = \\
&d_1^2 d_2^2 (r_2^2 s_{32}^2 - r_1^2 s_{31}^2) \,/\, r_1^2 r_2^2 \eta^2 = \\
&d_1^2 d_2^2 (s_{32}^2 / r_1^2 - s_{31}^2 / r_2^2) \,/\, \eta^2 = \\
&\qquad\qquad \text{(by Law of Sines)} \\
&d_1^2 d_2^2 (s_{12}^2 / r_3^2 - s_{21}^2 / r_3^2) \,/\, \eta^2 = \\
&d_1^2 d_2^2 (s_{12}^2 - s_{21}^2) \,/\, r_3^2 \eta^2 = \\
&d_1^2 d_2^2 (c_{21}^2 - c_{12}^2) \,/\, r_3^2 \eta^2 = \\
&\qquad\qquad \text{(by Lemma 2)} \\
&4 r_1^2 r_2^2 (c_{21}^2 - c_{12}^2) \,/\, d_3^2 z^2.
\end{aligned}$$

□

Lemma 10.

$$\lim_{|z|\to\infty} \frac{d_1^2 s_2^2 - d_2^2 s_1^2}{\eta^2} = \frac{4\,(r_2^2 c_{21}^2 - r_1^2 c_{12}^2)}{d_3^2} = \frac{4\,(e_2^2 - f_1^2 - e_1^2 + f_2^2)}{d_3^2} .$$

Proof. By Lemma 9, $(d_1^2 s_2^2 - d_2^2 s_1^2) \,/\, \eta^2 = 4 r_1^2 r_2^2 (c_{21}^2 - c_{12}^2) \,/\, d_3^2 z^2$. Now, $4 r_1^2 r_2^2 c_{21}^2 \,/\, d_3^2 z^2 = 4 r_1^2 (e_2^2 - f_1^2) \,/\, d_3^2 z^2$, and this approaches $4(e_2^2 - f_1^2) \,/\, d_3^2$, since $r_1^2 / z^2 \to 1$. Likewise, $4 r_1^2 r_2^2 c_{12}^2 \,/\, d_3^2 z^2$ approaches $4(e_1^2 - f_2^2) \,/\, d_3^2$. So the claim follows.

□

Lemma 11.

$$z^2 \left[\frac{d_1^2 s_2^2 - d_2^2 s_1^2}{\eta^2} - \lim_{|z|\to\infty} \frac{d_1^2 s_2^2 - d_2^2 s_1^2}{\eta^2} \right]$$

does not depend on z, and equals

$$\frac{4\,(e_2^2 f_2^2 - e_1^2 f_1^2)}{d_3^2} .$$

Proof. By the previous two lemmas, the quantity in question equals

$$\begin{aligned}
&z^2 \left[\frac{4 r_1^2 r_2^2 (c_{21}^2 - c_{12}^2)}{d_3^2 z^2} - \frac{4\,(r_2^2 c_{21}^2 - r_1^2 c_{12}^2)}{d_3^2} \right] = \\
&\frac{4\left[(r_1^2 - z^2)(r_2^2 c_{21}^2) - (r_2^2 - z^2)(r_1^2 c_{12}^2)\right]}{d_3^2} = \\
&\frac{4\left[e_1^2 (e_2^2 - f_1^2) - e_2^2 (e_1^2 - f_2^2)\right]}{d_3^2} = \\
&\frac{4\,(e_2^2 f_2^2 - e_1^2 f_1^2)}{d_3^2} .
\end{aligned}$$

□

4.2 Proof of the Principal Theorem

We are now ready to rapidly establish the principal theorem in this article.

Proof of Theorem 1. By Lemmas 1, 7, 8, 10 and 11,

$$\frac{d_1^2 s_2^2 - d_2^2 s_1^2}{\eta^2} =$$

$$\frac{4\,(e_2^2 - f_1^2 - e_1^2 + f_2^2)}{d_3^2} - \frac{4\,(e_1^2 f_1^2 - e_2^2 f_2^2)}{d_3^2\, z^2} =$$

$$\csc\left(\frac{\phi_1-\phi_2}{2}\right)\Sigma(\psi;\,x+x_3,y+y_3)+$$

$$\frac{1-x^2-y^2}{z^2}\left[\frac{d_1^2-d_2^2}{4}-\right.$$

$$\left.\csc\left(\frac{\phi_1-\phi_2}{2}\right)\Sigma(\psi;\,x-\frac{x_1+x_2}{2},y-\frac{y_1+y_2}{2})\right],$$

where $\psi=(\phi_1+\phi_2+2\phi_3)/2$.

□

5 AN APPLICATION TO CAMERA TRACKING

5.1 A special setup

In anticipation of an iterative P3P method to be introduced and studied in this section, it will be necessary to restrict attention to the special setup where the control points are such that $t_1+t_2+t_3=0$. However, this is actually *not* a serious restriction because it can always be achieved by simply rotating the coordinate system about the z-axis:

Imagine rotating by an amount α, and thus changing ϕ_j to $\phi_j-\alpha$ for $j=1,2,3$. If we let $u=\tan(\alpha/2)$, then $t_j=\tan(\phi_j/2)$ gets replaced by $\tan(\phi_j/2-\alpha/2)=(t_j-u)/(1+t_ju)$. Thus $t_1+t_2+t_3$ is replaced by $(t_1-u)/(1+t_1u)+(t_2-u)/(1+t_2u)+(t_3-u)/(1+t_3u)$. Upon setting this to zero and multiplying through by the common denominator, we obtain a cubic equation in u. Solving for u then allows for the determination of an angle of rotation α that will result in the desired special setup.

Henceforth we will assume that we are working in this special set up; that is, we will assume that $t_1+t_2+t_3=0$. Now, define the function

$$f(\phi_1,\phi_2,\phi_3;x,y,W)\;=\;(t_1-t_2)(1+t_3^2)\cdot$$
$$[A(\phi_1,\phi_2,\phi_3;x,y)+B(\phi_1,\phi_2,\phi_3;x,y)\,W],$$

with A and B as in Theorem 1. This theorem provides a way to compute $f(\phi_1,\phi_2,\phi_3;x,y,W)$, $f(\phi_2,\phi_3,\phi_1;x,y,W)$ and $f(\phi_3,\phi_1,\phi_2;x,y,W)$, when $W=(1-x^2-y^2)/z^2$, solely from a knowledge of t_1,t_2,t_3,c_1,c_2, and c_3, values which are presumed known a priori in P3P. Some related functions turn out to be handier to work with though. These are defined as follows:

$$g(\phi_1,\phi_2,\phi_3;x,y,W)\;=\;[\,f(\phi_1,\phi_2,\phi_3;x,y,W)+$$
$$f(\phi_2,\phi_3,\phi_1;x,y,W)\,+\,f(\phi_3,\phi_1,\phi_2;x,y,W)\,]\,/\,3\quad\text{and}$$

$$h(\phi_1,\phi_2,\phi_3;x,y,W)\;=$$
$$[\,t_3\,f(\phi_1,\phi_2,\phi_3;x,y,W)\,+\,t_1\,f(\phi_2,\phi_3,\phi_1;x,y,W)+$$
$$t_2\,f(\phi_3,\phi_1,\phi_2;x,y,W)\,]\,/\,(t_1^2+t_2^2+t_3^2).$$

When $W=(1-x^2-y^2)/z^2$, these two functions can also be computed directly from t_1,t_2,t_3,c_1,c_2, and c_3. But they are also conveniently expressed according to the following corollary. The proofs of the claims here are simply a matter of expanding the expressions involved and collecting the coefficients of x, y and W, remembering that $t_1+t_2+t_3=0$.

Corollary 3. *Let $t_\pi=t_1t_2t_3$ and $t_\sigma=t_1^2+t_2^2+t_3^2$. Then*

$$g(\phi_1,\phi_2,\phi_3;x,y,W)\;=\;t_\pi(1-W)(x^2-y^2)+$$
$$(2+t_\sigma)(1-W)xy+2t_\pi(1-2W)x+$$
$$[2(1+W)+t_\sigma(1-W)]y-3t_\pi(1+W),\ \textit{and}$$

$$h(\phi_1,\phi_2,\phi_3;x,y,W)\;=\;(2+t_\sigma)(1-W)(y^2-x^2)/2$$
$$+2t_\pi(1-W)xy+(2+t_\sigma-4W)x$$
$$-2t_\pi(1+W)y+(6-t_\sigma)(1+W)/2.$$

When $P=(x,y,z)$ is on the danger cylinder, so that $W=(1-x^2-y^2)/z^2=0$, the above formulas becomes simpler. In this case, using Weierstrass' substitution, $x=(1-t^2)/(1+t^2)$, $y=2t/(1+t^2)$, for an unknown t, we obtain two fourth degree polynomial equation in t. By repeatedly using these and eliminating higher powers of t, it is possible to reduce to a linear equation in t with known coefficients. The derivation of this is too tedious to present here, though it just amounts to manipulating polynomials as is done when directly computing a resultant polynomial. The resulting rational formula for t can be found in Appendix C, though "t" here is "t'" there. Also, "μ" and "ν" there are the computed value of the functions g and h, respectively, based on the values of t_1,t_2,t_3,c_1,c_2, and c_3.

This rational formula for t has been very successful in computer simulations. It does produce the correct value for t, and so the correct values for x and y

as well, when P is on the danger cylinder. Even when P is only near to the danger cylinder, it can be used to find a point on the danger cylinder that is close to P. This useful fact will be exploited in an iterative algorithm to be developed in this section.

Once x and y have been determined, system (2) can be used to find the value of each $r_j^2 - z^2$. Substituting into (1) then produces three equations in just one unknown, namely z^2. These can be manipulated to produce quadratic polynomial equations in z^2, and then the z^4 terms can be eliminated. Three linear equations in z^2 can be produced in this manner. Appendix C has a formula for z^2 (called "Z'" there) based on these.

5.2 An iterative algorithm for P3P-based camera tracking

Camera tracking based on three control points (ignoring orientation) is traditionally regarded as a matter of solving systems (1) and (2) algebraically, and then somehow trying to select the actual solution from among up to four mathematical solutions. Several different approaches to solving (1) have been developed over the years (Hanning et al., 2006). These all depend on being able to solve a quartic (*i.e* fourth degree) polynomial.

While these approaches are generally capable of finding all the mathematical solution to (1), there are several drawbacks for dynamic camera tracking using them. With four solutions to choose from, there is no surefire way to know which of these is the physically correct solution. Knowing the camera's recent position and perhaps its recent movement can help to more reliably choose the correct solution. However, even with accurate history information, errors are particularly likely near the danger cylinder, where nearly equal solutions occur. Clearly, as soon as an error occurs, it is likely to propagate into the future, since the history information would now become corrupt.

Moreover, the computational time and power needed to solve the quartic equation might be prohibitive in a realtime situation with only modest processing power. As pointed out in (Rieck, 2011), solving a quartic equation whose coefficients and roots are all real numbers might nevertheless demand arithmetic involving complex numbers, placing an additional requirement on the computational environment.

An iterative tracking method will now be proposed. The details are given in the pseudocode in Appendix A and in the subsequent appendices. The algorithm requires only real-number arithmetic, and uses only the basic arithmetic operations and square root operation. Unlike the "algebraic methods" just discussed, it does not normally require selecting from multiple mathematical solutions. The only exception to this is near the danger cylinder, where the number of potential solution options is limited to only two, rather than four. Moreover, such a selection is only necessary when moving away from the danger cylinder vicinity.

The new algorithm begins with a basic iterative method (Appendix B) that is easy to understand and derive, and which, in simulations, has been found to accurately track a not-too-quickly moving camera that is not too close to the danger cylinder. One simply uses a multivariate Newton-Raphson method on the system (1) in order to estimate the change in (r_1, r_2, r_3) based on a knowledge of the change in (c_1, c_2, c_3).

The Jacobian matrix J for this system, discussed in Subsection 2.5, is straightforward to compute and invert. However, it is not invertible on the danger cylinder, so the basic method breaks down there. When this is not the case, one simply uses the approximation

$$\begin{bmatrix} \Delta r_1 \\ \Delta r_2 \\ \Delta r_3 \end{bmatrix} \approx J^{-1} \begin{bmatrix} \Delta c_1 \\ \Delta c_2 \\ \Delta c_3 \end{bmatrix}$$

and the previous estimates for the r_j to obtain updated values for these. Once the r_j have been estimated, system (2) can be solved for (x, y, z).

When the camera gets too close to the danger cylinder, there are two related difficulties. Firstly, as suggested by Proposition 1, based on the measured cosine angles c_j, there will be multiple nearby mathematical solutions to systems (1) and (2), since repeated solutions occur on the danger cylinder. Typically (in fact almost always), there will just be two such solutions, one of which is the physically correct solution. Since the solutions are close together, it is quite likely that the basic iterative method will accidentally converge to the wrong solution.

Secondly, this iterative method involves dividing by the determinant of J. Since this gets small near the danger cylinder, computational difficulties leading to inaccuracies arise. To mitigate this, one can replace a too small J by some fixed quantity with the same sign as J (±JCLIP). However, this slows down the convergence rate for the algorithm.

Since the `basic-iteration` method runs into difficulties near the danger cylinder, and could accidentally end up latching onto an incorrect solution, one might be inclined to switch to one of the algebraic methods in this case. However, besides being complicated and requiring complex arithmetic, these methods also have a problem with accuracy near the danger cylinder. Essentially the same singularity issue causes substantial roundoff error.

Moreover, while the camera estimate continues to wander near the danger cylinder, it would be necessary after each iteration to select the correct solution from among multiple mathematical solutions that are close together. Even with a good selection strategy, the probability becomes high that a mistake would be made during some selection.

In light of all these issues near the danger cylinder, a new variation on the above iterative algorithm will now be introduced, one that modifies the earlier updating process for estimating the position near the danger cylinder. The essential philosophy of this alteration to the basic iterative method is that *it is time-consuming, futile and even dangerous to try to accurately track the camera when it is too close to the danger cylinder. One should instead track a point on the danger cylinder that is close to the camera.*

The new algorithm senses when the camera estimate wanders close to the danger cylinder (within a distance `CLOSE`). When this happens, rather than simply relying on `basic-iteration`, hoping to find the intended solution, it instead first locates a nearby point on the danger cylinder. This can be rapidly accomplished in light of the previous subsection, using the method `lock-onto-cylinder` in Appendix C. Note that this method does not depend on the previously estimated position, but only on the measured cosine data and the fixed control point data.

The cosines c_j corresponding to this point are next checked to see if they are reasonably well matched with the measured cosines. Specifically, a test is made to see if the sum of the squared differences of the cosines is within a fixed tolerance `COSTOL`. If not, the algorithm reverts back to the `basic-iteration` method. But if so, then the algorithm moves into the "locked" (on the danger cylinder) state and uses the point on the cylinder as the estimated camera position.

It then remains locked onto the danger cylinder during subsequent iterations of the method, until it detects that the cosines corresponding to the danger cylinder point are a poor match for the measured cosines of the actual camera position. While in the locked condition, the camera position estimate will always be a point on the danger cylinder, found via the method in Appendix C.

When a poor cosines match occurs, another method is needed to find two nearby points, off the danger cylinder, that tend to better match the measured cosines. It is then necessary to decide which of these has the better chance of being closer to the physical solution, *i.e.* the camera's actual position. This article offers no comment on how to make this decision, but it is a serious issue for all P3P tracking methods that must select between multiple mathematical solutions. Work remains to be done concerning this. Here at least there are only two solutions to choose between, as opposed to potentially four solutions produced by the algebraic methods.

A method `exit-cylinder` has been developed to produce two such nearby points (off the cylinder), but the exact details for it are not included in this article, and will be presented elsewhere. This method does a good job when the camera is not moving too fast, as will be demonstrated in the next subsections. A very brief explanation of the method's derivation is as follows.

Consider the prior estimate point (x,y,z) on the cylinder, with associated values r_1,r_2,r_3. Consider also the error in these values and those of the camera's actual position, off the cylinder. Up to third order, there exists a certain linear combination of Δr_1, Δr_2 and Δr_3, with coefficients involving r_1, r_2, r_3, d_1, d_2 and d_3, whose square equals κ, another quantity that can be computed from r_1, r_2, r_3, d_1, d_2, d_3, Δc_1, Δc_2 and Δc_3. The linear combination of Δr_1, Δr_2 and Δr_3 is thus approximately equal to $\pm\sqrt{\kappa}$. Two other independent linear (approximate) equations in Δr_1, Δr_2 and Δr_3 can be obtained from

$$\begin{bmatrix} \Delta c_1 \\ \Delta c_2 \\ \Delta c_3 \end{bmatrix} \approx J \begin{bmatrix} \Delta r_1 \\ \Delta r_2 \\ \Delta r_3 \end{bmatrix} .$$

Together these produce a linear system of (approximate) equations that can be solved for $(\Delta r_1,\Delta r_2,\Delta r_3)$. Of course, the solution depends of the sign choice in $\pm\sqrt{\kappa}$. Hence there are actually two mathematical solutions for $(\Delta r_1,\Delta r_2,\Delta r_3)$.

Although the computations for the overall tracking algorithm are somewhat involved, they are nevertheless direct. Away from the danger cylinder, where the `basic-iteration` method is suitable, the number of arithmetic operations is certainly less than the number of operations needed to set up and solve the quartic equation that occurs in any of the algebraic methods. As long as the camera is moving at an acceptably slow speed, the overall iterative method, with accommodation for the danger cylinder, behaves quite well, as will be seen.

Other iterative methods for solving P3P would probably also benefit from a similar adjustment when the camera is near the danger cylinder, partly because there would always be the possibility of converging to the wrong solution, near the danger cylinder. These might include the Gauss-Newton approach used by (Fung and Wong, 1998), as well as the method in Appendix A.3 of (Fischler and Bolles, 1981) that repeatedly relocates one of the control points and determines the two possible locations of each of the other two control points.

Eventually it would be worthwhile having a comparison of a wide variety of techniques for P3P tracking. However, comparisons with the algebraic methods are complicated by the need to choose criteria for selecting a solution from among the various mathematical solutions. As already indicated, the issue of making such choices is not addressed in the present article. Iterative techniques avoid this issue, but generally break down near the danger cylinder, unless somehow modified, possibly along the lines presented above.

5.3 Simulations

The P3P-based tracking algorithm, as introduced in the previous section and detailed in the appendices, has been implemented using Mathematica® [1]. The computations involve 64-bit floating point data. Recall that the danger cylinder radius is set to one throughout this article. This remains the case here. In the simulations, the three algorithm parameters were set as follows: `CLOSE` = 0.03, `COSTOL` = 0.000001 and `JCLIP` = 10^{-50}.

The simulations were conducted on three different control point configurations including one where these points were equidistant from each other. There was no significant difference in the results. For the remainder of this subsection, the specific results will be presented for the setup using $t_1 = 4.3$, $t_2 = -4.0$ and $t_3 = -0.3$, and hence $\phi_1 = 2.684$, $\phi_2 = -2.652$, $\phi_3 = -0.583$ (radians). Note that $t_1 + t_2 + t_3 = 0$ as required (see Subsection 5.1).

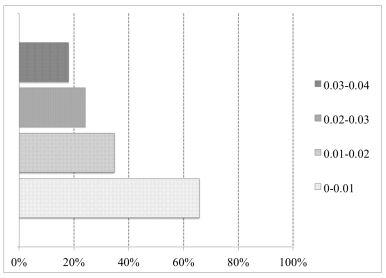

Figure 5: % lockings based on distance to cylinder

The simulations involved repeatedly randomly generating a current estimate (x, y, z) for the camera's position, referred to as the "previously estimated position," within a distance of 0.04 of the danger cylinder. Notice that the condition $|1 - \sqrt{x^2 + y^2}| >$ `CLOSE` in the algorithm was usually false. The simulated camera was then randomly placed at another point, also within a distance of 0.04 of the danger cylinder.

The new camera position was always selected to be randomly within a distance 0.1 (10% of danger cylinder radius) of the previously estimated position. The two positions were also chosen so that their distances to the plane containing the control points were randomly between 1 and 4. Once the points were chosen, a couple iterations of the algorithm's loop (see Appendix A) were executed, with the algorithm initially in the unlocked condition. This random exercise was repeated 500,000 times. The resulting data were then analyzed.

Figure 5 shows the percentage of times that the algorithm decided to lock onto the danger cylinder, as a function of the distance of the actual camera position to the danger cylinder. However, it only involves the cases for which the camera's previously estimated position was within a distance of 0.03 from the danger cylinder, making the condition $|1 - \sqrt{x^2 + y^2}| >$ `CLOSE` false. This means there was a chance of locking onto the cylinder, and this decision was made based on the cosine values.

[1] A Mathematica notebook with the simulation code and results is available from the author upon request. Equivalent C++ code can also be produced upon request.

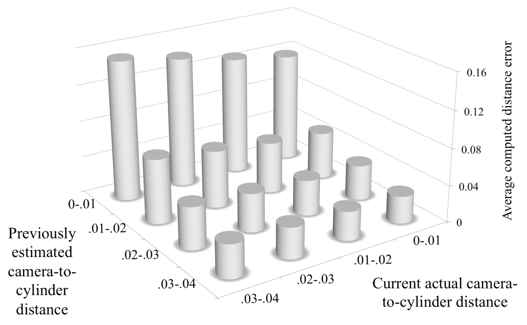

Figure 6: Average errors in unlocked cases

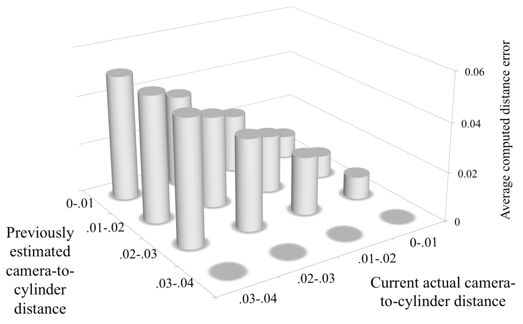

Figure 8: Average errors in locked cases

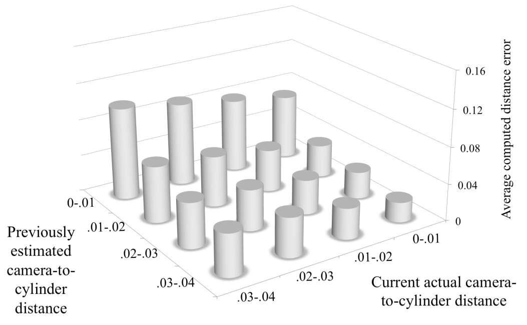

Figure 7: Average errors after two iterations

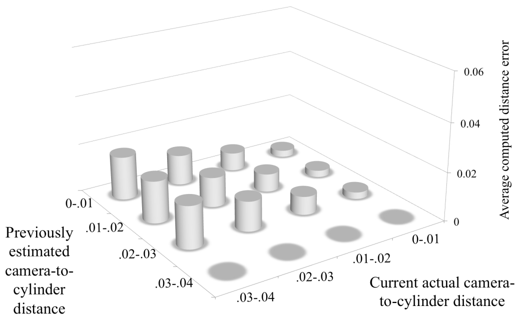

Figure 9: Average errors after leaving cylinder

5.4 Error Results

Figures 6 though 9 show average errors between the newly computed estimate of the camera's position and its actual new position, after applying one or two iterations of the new algorithm in Appendix A. Each figure shows this error as a function of both the previously estimated position and the new camera position.

Figure 6 shows the situation after a single iteration and deals only with the case when the iteration did not cause the algorithm to become locked onto the danger cylinder. We see that the errors were quite large when the previous estimates were too close to the danger cylinder. To help control these errors, the value of `COSTOL` could be increased, as discussed below.

Figure 7 shows what happened in the same situation when, after the first iteration, the camera stopped moving and a second iteration of the algorithm was applied. Clearly there was improvement, but substantial errors remained when the position estimation began too close to the danger cylinder.

Figure 8 is similar to Figure 6, but it reflects the situation where the iteration caused the algorithm to transition from the unlocked state to the locked state. (Notice that the vertical axes for Figures 8 and 9 are scaled differently then those of Figures 6 and 7.) Since the newly estimated positions were always on the danger cylinder in this case, it is not surprising that the error grew as a function of the distance of the actual new camera position to the danger cylinder.

Figure 9 reflects what happened, after the locked situation in Figure 8, when the camera was suddenly moved away from the danger cylinder, precipitating the need to unlock and move the estimated position off of the danger cylinder. To effect this, the `exit-cylinder` method was used to find two points near the estimated point on the danger cylinder.

From these two points, the one that was closer to the actual camera position was selected. This selection is not realistic in practice since the actual camera position is unknown in a real-world situation. However, as already discussed, solution selection continues to be a thorny issue common to P3P methods. This article is not focused on this important issue. Figure 9 at least shows that when the selection between the two options here is correctly made, the results can be highly satisfactory.

Increasing the value of `COSTOL` makes it more likely that locking onto the danger cylinder will occur. When `COSTOL` was set to 0.0005 instead of 0.000001, the likelihood of locking went up to 88% when the camera was within a distance 0.01 of the danger cylinder, while the average distance error when this happened was 0.014. When the camera-to-danger-cylinder distance was between 0.03 and 0.04, the likelihood of locking was 57%, and the average distance error when this happened was 0.075. Deciding when to lock and how to compute a point on the cylinder are both areas of future study that could significantly reduce errors.

Although there is not much data presented here in the $|1-\sqrt{x^2+y^2}| >$ `CLOSE` case, the basic method (Appendix B) performs quite well in the tested setups. The trend seen in Figure 6, as the previously estimated position gets further from the danger cylinder, continued further out, with the average errors continuing to drop off. The collective data analysis reveals that over a broad range of distances from the danger cylinder and a broad range of distances from the control points, usable position estimations can be expected from the overall algorithm when the camera is allowed to move up to ten percent of the danger cylinder radius per iteration.

Grunert's algebraic method was also implemented, as specified in his original paper (Grunert, 1841). A very simple criterion was used to select from among the multiple mathematical solutions produced: the one closest to the previously estimated position was always selected. In general, this worked well, and outperformed the basic iterative method. However, as expected, Grunert's method also suffered significantly near the danger cylinder. Therefore, an altered version of this method was also produced, one that implemented locking onto the danger cylinder under the same nearness conditions that were employed in the altered version of the iteration method. Near the danger cylinder, the average distance error in tracking the correct solution improved with the addition of this locking mechanism.

Experiments were conducted on the two versions of Grunert's method. The parameters were the same as in the earlier iterative method experiments, except that the camera movement was kept within a distance of 0.5 of the previously estimated position. When the previously estimated camera position was within 0.01 of the danger cylinder, and the new camera position was within 0.02 of the danger cylinder, Grunert's method without locking had an average error of 0.04, while the locking mechanism reduced this error to 0.01. It should be noted though that the more important potential benefit of adding a "locking onto the danger cylinder" feature is the same for all P3P methods: it can allow for the postponement of tricky decision making until the camera has moved sufficiently far away from the danger cylinder.

5.5 Timing Results

The addition of the `lock-onto-cylinder` and `exit-cylinder` methods to the basic method certainly does create a more complicated algorithm. But as can be seen in the figures, this results in a much improved algorithm, in the vicinity of the danger cylinder. Additionally, the cost in terms of computation time is not too bad. Glancing at the appendices reveals that the number of computations involved in using the `lock-onto-cylinder` method is not worse than about two or three times the number of computations involved in the using the basic method. The number of computations involved in the `exit-cylinder` method is somewhat worse, by a factor of about two.

When the Mathematica code was compiled (instead of interpreted), it became possible to conduct accurate timing experiments, on a 2.4GHz MacBook Pro. The results were quite encouraging, especially in light of the somewhat complicated calculations.

Attention was paid to the overhead time involved in making calls to the (completely) compiled code for the overall algorithm. This time was subtracted off measured times for single iteration executions. Thus the numbers reported here are the average CPU times for executing a single iteration of the algorithm, and only this code. The averages were based on 5000 carefully examined trials.

When one iteration was used to update the position estimate, beginning in the unlocked state and with $|1-\sqrt{x^2+y^2}| >$ `CLOSE`, and hence remaining in the unlocked state, the average execution time was 22.48 microseconds. This would be essentially the same time required for just using the basic update method, without testing for danger cylinder nearness, since nothing else was involved in this case except this brief test.

There is of course another case in which algorithm remains in the unlocked state, as follows. When $|1-\sqrt{x^2+y^2}| \leq$ `CLOSE`, a point on the cylinder is computed using the `lock-onto-cylinder` method. But if this point yields a poor cosines match with the measured cosines, then this point is rejected, and the algorithm invokes the basic method to update the position estimate, and remains unlocked. All of this required, on average, 47.19 microseconds.

The case where an iteration caused a transition from the unlocked state to the locked state was quite surprising. This required failing the $|1-\sqrt{x^2+y^2}| >$ `CLOSE` test, finding the point on the cylinder, and then verifying that it is a good cosines match, and so locking onto it. This took only 23.36 microseconds, on average.

The case where a transition occurred from the locked state to the unlocked state required 93.89 microseconds, on average. In this case, a new point on the cylinder needed to be calculated, which then must have failed to be a good cosines match, causing the `exit-cylinder` method to be invoked. Hopefully some way will eventually be discovered to improve this time, which is more than four times as long as the `basic-iteration` time. Still, the time penalty here needs to be considered against the likelihood of an error when using the basic iterative method, or this likelihood and/or time penalty for other methods.

Grunert's method, when compiled in Mathematica, took quite a long time to execute, nearly two milliseconds. This was so regardless of whether `Solve` or `NSolve` was used to solve the quartic equation. However, one of the referees of this paper was able solve the quartic equation and thence solve (1), using Matlab on a standard MacBook, in 74 microseconds. This of course is much more in line with the times indicated above for the iterative method.

6 CONCLUSION

Several results in plane geometry were presented. These were used as cornerstones for the principal result of this article, which is a theorem in solid geometry. The theorem was used to reformulate the classic P3P problem, by producing a very different system of equations, with a very explicit connection to the danger cylinder.

This new framework was then used to enhance the behavior near the danger cylinder of a simple iterative technique for finding approximate solutions to P3P. This enhancement involved the addition of a new "lock onto cylinder" technique whenever the camera is detected to be sufficiently close to the danger cylinder. This technique can also be adapted to other P3P methods, to better accommodate the inherent unpredictability caused by being in the proximity of the danger cylinder.

Further work should be done to fine-tune parameters and make other helpful adjustments to the locking technique. There is reason to hope that simpler formulas and improved performance will result. There is also the expectation that the new framework for P3P will lead to further useful developments in the understanding of this problem near the danger cylinder, resulting in other useful applications for camera tracking.

REFERENCES

DeMenthon, D. and Davis, L. S. (1992). Exact and approximate solutions of the perspective-three-point problem. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 14(11):1100–1105.

Faugère, J.-C., Moroz, G., Rouillier, F., and El-Din, M. S. (2008). Classification of the perspective-three-point problem, discriminant variety and real solving polynomial systems of inequalities. In *ISSAC'08, 21st Int. Symp. Symbolic and Algebraic Computation*, 79–86.

Fischler, M. A., Bolles, R. C. (1981). Random sample consensus: a paradigm for model fitting with applications to image analysis and automatic cartography. In *Graphics and Image Processing*, 24(6): 381–395.

Fung, Y. F., Wong, K. H. (1998). A three-point model-based algorithm for pose estimation. *1998 Int. Symp. Image, Speech, Signal Processing and Robotics (ISSPR'98).*

Gao, X.-S., Hou, X.-R., Tang, J., and Cheng, H.-F. (2003). Complete solution classification for the perspective-three-point problem. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 25(8): 930–943.

Grunert, J. A. (1841). Das pothenotische problem in erweiterter gestalt nebst über seine anwendungen in der geodäsie. In *Grunerts Archiv für Mathematik und Physik*, 1: 238–248.

Hanning, T., Schöne, R., and Graf, S. (2006). A closed form solution for monocular re-projective 3d pose estimation of regular planar patterns. In *IEEE Int. Conf. Image Processing (ICIP)*, 2197–2200.

Haralick, R. M., Lee, C.-N., Ottenberg, K., and Nölle, N. (1994). Review and analysis of solutions of the three point perspective pose estimation problem. *J. Computer Vision*, 13(3): 331–356.

Merritt, E. L. (1949). Explicit three-point resection in space. *Photogrammetric Engineering*, 15(4): 649–655.

Müller, F. J. (1925). Direkte (exakte) lösung des einfachen rückwärtsein-schneidens im raume. In *Allegemaine Vermessungs-Nachrichten*.

Nistér, D. (2007). A minimal solution to the generalized 3-point pose problem. *J. Mathematical Imaging and Vision*, 27(1): 67–79.

Pisinger, G. and Hanning, T. (2007). Closed form monocular re-projection pose estimation. In *ISIP '07, IEEE Int. Conf. Image Processing*, 5: 197–200.

Rieck, M. Q. (2010). Handling repeated solutions to the perspective three-point pose problem. In *VISAPP '10,* *Int. Conf. Computer Vision Theory and Appl.*, 1: 395–399.

Rieck, M. Q. (2011). An algorithm for finding repeated solutions to the general perspective three-point pose problem. *J. Mathematical Imaging and Vision*. 42(1): 92–100.

Rieck, M. Q. (2012). Solving the Three-Point Camera Pose Problem in the Vicinity of the Danger Cylinder. In *VISAPP '12, Int. Conf. Computer Vision Theory and Appl.*, 2: 335-340.

Smith, A. D. N. (1965). The explicit solution of the single picture resolution problem, with a least squares adjustment to redundant control. *Photogrammetric Record*, 5(26): 113–122.

Sun, F.-M. and Wang, B. (2010). The solution distribution analysis of the p3p problem. In *SMC '10, Int. Conf. Systems, Man and Cybernetics*, 2033–2036.

Tang, J., Chen, W., and Wang, J. (2008). A study on the p3p problem. In *ICIC '08, 4th Int. Conf. Intelligent Computing*, 5226: 422–429.

Tang, J. and Liu, N. (2009). The unique solution for p3p problem. In *SIGAPP '09, ACM Symp. Applied Computing*, 1138–1139.

Thompson, E. H. (1966). Space resection: failure cases. *Photogrammetric Record*, 5(27): 201–204.

Wolfe, W. J., Mathis, D., Sklair, C. W., and Magee, M. (1991). The perspective view of three points. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 13(1): 66–73.

Xiaoshan, G. and Hangfei, C. (2001). New algorithms for the perspective-three-point problem. *J. Comput. Sci. & Tech.*, 16(3): 194–207.

Zhang, C.-X. and Hu, Z.-Y. (2005). A general sufficient condition of four positive solutions of the p3p problem. *J. Comput. Sci. & Technol.*, 20(6): 836–842.

Zhang, C.-X. and Hu, Z.-Y. (2006). Why is the danger cylinder dangerous in the p3p problem? *Acta Automatica Sinica*, 32(4): 504–511.

APPENDICES

A Iterative Algorithm

The following is pseudocode for the camera tracking algorithm introduced in Subsection 5.2. Prior to tracking, the values t_j, x_j, y_j and d_j ($j = 1,2,3$) for the control points are obtained (with $t_1 + t_2 + t_3 = 0$), and the values $t_\pi = t_1 t_2 t_3$, $t_\sigma = t_1^2 + t_2^2 + t_3^2$ and $\iota = 2(x_1 y_2 + x_2 y_3 + x_3 y_1 - x_2 y_1 - x_3 y_2 - x_1 y_3)$ are calculated.

An estimate (x,y,z) of the camera's position is maintained and updated with each iteration of the following loop. Along with this, corresponding distances to the control points (r_1, r_2, r_3), as in system (2), and matching cosines, as in system (1), are also maintained and updated. Each iteration also entails using up-to-date camera images and the cosines $\hat{c}_1, \hat{c}_2, \hat{c}_3$ implied by these images.

```
Set LOCKED = false.
Repeat (while tracking):
  Measure cosines $\hat{c}_1, \hat{c}_2, \hat{c}_3$ for new camera
    position.
  If not LOCKED and $|1 - \sqrt{x^2+y^2}| >$ CLOSE
  then
    Use basic-iteration method to set
      estimated camera position $(x,y,z)$ by
      changing it to returned value $(x',y',z')$.
    Likewise set $r_1, r_2, r_3, c_1, c_2, c_3$ (distances, cosines).
  else
    Use lock-onto-cylinder method to find
      a point $(x',y',z')$ on the danger cylinder
      close to previously estimated position $(x,y,z)$.
    Also obtain corresponding $r_1', r_2', r_3', c_1', c_2', c_3'$.
    If $(c_1' - \hat{c}_1)^2 + (c_2' - \hat{c}_2)^2 + (c_3' - \hat{c}_3)^2 >$ COSTOL
    then
      If LOCKED
      then
        Use exit-cylinder method to set
          estimated camera position $(x,y,z)$ by
          changing it to returned value $(x',y',z')$.
        Likewise set $r_1, r_2, r_3, c_1, c_2, c_3$.
      else
        Use basic-iteration method to set
          estimated camera position $(x,y,z)$ by
          changing it to returned value $(x',y',z')$.
        Likewise set $r_1, r_2, r_3, c_1, c_2, c_3$.
      End-If
      Set LOCKED = false.
    else
      Set $(x,y,z)$ to $(x',y',z')$, and likewise set
        $r_1, r_2, r_3, c_1, c_2, c_3$.
      Set LOCKED = true
    End-If
  End-If
End-Repeat
```

The `basic-iteration` method is detailed in Appendix B. The `lock-onto-cylinder` method is detailed in Appendix C. These methods are also discussed in Section 5.2. The `exit-cylinder` method is outlined briefly in Section 5.2, and will be explained in more detail elsewhere. Details can also be found in the available Mathematica and C++ source code.

B Basic-Iteration Method

Input: control points data $(d_1, d_2, d_3, x_1, x_2, x_3, y_1, y_2, y_3, \iota)$, previously estimated position and distance data (x, y, z, r_1, r_2, r_3), and currently measured cosine data $(\hat{c}_1, \hat{c}_2, \hat{c}_3)$. (See Appendix A.)

Output: updated estimated position, distance and matching cosine data $(x', y', z', r_1', r_2', r_3', c_1', c_2', c_3')$.

◇ Set $R_j = r_j^2$ and $D_j = d_j^2$ for $j = 1,2,3$.

◇ Set $J = D_1D_2D_3 - D_1R_1^2 - D_2R_2^2 - D_3R_3^2 + (D_1 + D_2 - D_3)R_1R_2 + (D_2 + D_3 - D_1)R_2R_3 + (D_3 + D_1 - D_2)R_3R_1$. However, if $|J| <$ JCLIP, then reset J to equal sign(J) JCLIP.

◇ Set $\Delta_0 r_3 = \frac{1}{2} r_3 [D_1D_2D_3 - D_3(D_1 + D_2)R_3 + D_1(D_2 - D_3)R_1 + D_2(D_1 - D_3)R_2 + D_3R_3^2 - D_1R_1^2 - D_2R_2^2 + (D_3 - D_1 - D_2)(R_3R_1 + R_3R_2 - 3R_1R_2) + R_3R_1(R_3 + R_1) + R_3R_2(R_3 + R_2) + R_1R_2(R_1 + R_2) - 6R_1R_2R_3] + r_2(D_3 - R_1 + R_2)R_3(D_2 + R_1 - R_3)\hat{c}_1 + r_1(D_3 + R_1 - R_2)R_3(D_1 + R_2 - R_3)\hat{c}_2 - r_1r_2r_3(D_2 + R_1 - R_3)(D_1 + R_2 - R_3)\hat{c}_3$, and similarly for $\Delta_0 r_1$ and $\Delta_0 r_2$ (by cycling subscripts).

◇ Set $r'_j = r_j + \Delta_0 r_j / J$ and $R_j = r_j^2$ for $j = 1,2,3$.

◇ Set $x' = [\,(x_3^2 + y_1y_2)(y_1 - y_2) + (x_1^2 + y_2y_3)(y_2 - y_3) + (x_2^2 + y_3y_1)(y_3 - y_1) + (y_3 - y_2)R'_1 + (y_1 - y_3)R'_2 + (y_2 - y_1)R'_3\,] / \iota$.

◇ Set $y' = [\,(y_3^2 + x_1x_2)(x_2 - x_1) + (y_1^2 + x_2x_3)(x_3 - x_2) + (y_2^2 + x_3x_1)(x_1 - x_3) + (x_2 - x_3)R'_1 + (x_3 - x_1)R'_2 + (x_1 - x_2)R'_3\,] / \iota$.

◇ Set $Z' = \frac{1}{3}[\,R'_1 + R'_2 + R'_3 - x_1^2 - x_2^2 - x_3^2 - y_1^2 - y_2^2 - y_3^2 + 2x'(x_1 + x_2 + x_3) + 2y'(y_1 + y_2 + y_3)) - (x'^2 + y'^2)\,]$.

◇ Set $z' = \sqrt{Z'}$.

◇ Compute cosine data c'_1, c'_2, c'_3 corresponding to r'_1, r'_2, r'_3.

C Lock-Onto-Cylinder Method

Input: control points data $(d_1, d_2, d_3, x_1, x_2, x_3, y_1, y_2, y_3, t_\pi, t_\sigma)$, and currently measured cosine data $(\hat{c}_1, \hat{c}_2, \hat{c}_3)$. (See Appendix A.)

Output: updated estimated position, distance and matching cosine data $(x', y', z', r'_1, r'_2, r'_3, c'_1, c'_2, c'_3)$.

◇ Set $D_j = d_j^2$, $\hat{C}_j = \hat{c}_j^2$ and $\hat{S}_j = 1 - \hat{C}_j$ for $j = 1,2,3$.

◇ Set $\eta = 1 - \hat{C}_1 - \hat{C}_2 - \hat{C}_3 + 2\hat{c}_1\hat{c}_2\hat{c}_3$.

◇ Set $\mu = 4[\,(t_1 - t_2)(t_1 - t_3)(t_2 + t_3 - 2t_1)\,\hat{S}_1 / (1 + t_1^2) + (t_2 - t_3)(t_2 - t_1)(t_3 + t_1 - 2t_2)\,\hat{S}_2 / (1 + t_2^2) + (t_3 - t_1)(t_3 - t_2)(t_1 + t_2 - 2t_3)\,\hat{S}_3 / (1 + t_3^2)\,] / (3\eta^2)$.

◇ Set $\nu = 4[\,(t_1 - t_2)(t_1 - t_3)(t_2^2 + t_3^2 - t_1t_2 - t_1t_3)\,\hat{S}_1 / (1 + t_1^2) + (t_2 - t_3)(t_2 - t_1)(t_3^2 + t_1^2 - t_2t_3 - t_2t_1)\,\hat{S}_2 / (1 + t_2^2) + (t_3 - t_1)(t_3 - t_2)(t_1^2 + t_2^2 - t_3t_1 - t_3t_2)\,\hat{S}_3 / (1 + t_3^2)\,] / (t_\sigma \eta^2)$.

◇ Set $t' = [\ 2\nu^3 t_\pi - \nu^2(\mu(t_\sigma + 2) + 8t_\pi(1 - t_\sigma)) + 8\mu^2 t_\pi(t_\sigma - 3) + 3\mu^3(t_\sigma + 2) - 32t_\pi(36t_\pi^2 + t_\sigma^2) - 4\mu(2t_\sigma^2 + t_\sigma^3 + 4t_\pi^2(t_\sigma + 30)) + \nu(10\mu^2 t_\pi + 8t_\pi(4(9t_\pi^2 - t_\sigma) + t_\sigma^2) - 4\mu(t_\sigma^2 + 2(t_\sigma - 14t_\pi^2)))\ ] / [\ 8\mu^3 t_\pi - 2\nu^3(t_\sigma + 2) + 48\mu t_\pi(18 + 4t_\pi^2 + 11t_\sigma + t_\sigma^2) - 4\nu^2(t_\pi^2 + 6t_\sigma + 3t_\sigma^2) + \mu^2(15t_\sigma^2 + 4(27 + 20t_\pi^2 + 21t_\sigma)) - 16(2t_\sigma^3 + t_\sigma^4 + 4t_\pi^2(t_\sigma^2 - 3(9 + 4t_\sigma))) - \nu(6\mu^2(2 + t_\sigma) + 84\mu t_\pi(2 + t_\sigma) + 8(6t_\sigma^2 + 3t_\sigma^3 + 4t_\pi^2(15 + 8t_\sigma)))\ ]$

◇ Set $x' = (1 - t'^2)/(1 + t'^2)$ and $y' = 2t'/(1 + t'^2)$.

◇ Set $a_j = x' - x_j$ and $b_j = y' - y_j$ for $j = 1,2,3$.

◇ Set $Z' = [\,(2a_1^2 + a_2^2 + a_3^2 + 2b_1^2 + b_2^2 + b_3^2 - D_2 - D_3)(a_2^2 - a_3^2 + b_2^2 - b_3^2 + D_2 - D_3)\hat{C}_1 - (a_1^2 + 2a_2^2 + a_3^2 + b_1^2 + 2b_2^2 + b_3^2 - D_1 - D_3)(a_1^2 - a_3^2 + b_1^2 - b_3^2 + D_1 - D_3)\hat{C}_2 + 4(a_1^2 - a_2^2 + b_1^2 - b_2^2)(a_3^2 + b_3^2)\hat{C}_1\hat{C}_2 + (a_1^2 + a_2^2 + 2a_3^2 + b_1^2 + b_2^2 + 2b_3^2 - D_1 - D_2)(a_1^2 - a_2^2 + b_1^2 - b_2^2 + D_1 - D_2)\hat{C}_3 - 4(a_2^2 + b_2^2)(a_1^2 - a_3^2 + b_1^2 - b_3^2)\hat{C}_1\hat{C}_3 + 4(a_1^2 + b_1^2)(a_2^2 - a_3^2 + b_2^2 - b_3^2)\hat{C}_2\hat{C}_3\,] / [\,4(a_2^2 - a_3^2 + b_2^2 - b_3^2 + D_2 - D_3)\hat{C}_1 - (a_1^2 - a_3^2 + b_1^2 - b_3^2 + D_1 - D_3)\hat{C}_2 + (a_1^2 - a_2^2 + b_1^2 - b_2^2)\hat{C}_1\hat{C}_2 + (a_1^2 - a_2^2 + b_1^2 - b_2^2 + D_1 - D_2)\hat{C}_3 - (a_1^2 - a_3^2 + b_1^2 - b_3^2)\hat{C}_1\hat{C}_3 + (a_2^2 - a_3^2 + b_2^2 - b_3^2)\hat{C}_2\hat{C}_3\,]$.

◇ Set $z' = \sqrt{Z'}$.

◇ Compute distance data r'_1, r'_2, r'_3 corresponding to point (x', y', z').

◇ Compute cosine data c'_1, c'_2, c'_3 corresponding to r'_1, r'_2, r'_3.