IMPROVEMENT OF PNP PROBLEM COMPUTATIONAL EFFICIENCY FOR KNOWN TARGET GEOMETRY OF CUBESATS

A Thesis

by

WILLIAM THOMAS HAFER

Submitted to the Office of Graduate Studies of
Texas A&M University
in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE

May 2012

Major Subject: Aerospace Engineering

Improvement of PNP Problem Computational Efficiency For Known Target Geometry of Cubesats

IMPROVEMENT OF PNP PROBLEM COMPUTATIONAL EFFICIENCY FOR KNOWN TARGET GEOMETRY OF CUBESATS

A Thesis

by

WILLIAM THOMAS HAFER

Submitted to the Office of Graduate Studies of
Texas A&M University
in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE

Approved by:

Chair of Committee,	Helen L. Reed
Committee Members,	John L. Junkins
	Jyh-Charn Liu
Head of Department,	Dimitris Lagoudas

May 2012

Major Subject: Aerospace Engineering

ABSTRACT

Improvement of PNP Problem Computational Efficiency For Known Target Geometry of Cubesats. (May 2012)

William Thomas Hafer, B.S., Massachusetts Institute of Technology

Chair of Advisory Committee: Dr. Helen L. Reed

This thesis considers the Perspective-N-Point (PNP) problem with orthogonal target geometry, as seen in the problem of cubesat relative navigation. Cubesats are small spacecraft often developed for research purposes and to perform missions in space at low cost. Sensor systems for cubesats have been designed that, by providing vector (equivalently line-of-sight, angle, and image plane) measurements, equate relative navigation to a PNP problem. Much study has been done on this problem, but little of it has considered the case where target geometry is known in advance, as is the case with cooperating cubesats. A typical constraint for cubesats, as well as other PNP applications, is processing resources. Therefore, we considered the ability to reduce processing burden of the PNP solution by taking advantage of the known target geometry. We did this by considering a specific P3P solver and a specific point-cloud correspondence (PCC) solver for disambiguating/improving the estimate, and modifying them both to take into account a known orthogonal geometry. The P3P solver was the Kneip solver, and the point-cloud-correspondence solver was the Optimal Linear

Attitude Estimator (OLAE). We were able to achieve over 40% reduction in the computational time of the P3P solver, and around 10% for the PCC solver, vs. the unmodified solvers acting on the same problems. It is possible that the Kneip P3P solver was particularly well suited to this approach. Nevertheless, these findings suggest similar investigation may be worthwhile for other PNP solvers, if (1) processing resources are scarce, and (2) target geometry can be known in advance.

ACKNOWLEDGEMENTS

Mom and dad, thank you for helping with my education, and doing all of the things that you did, to convince me that math is pretty easy.

Dr. Reed, thank you for giving me an unparalleled opportunity to pursue my education; and for your continued close collaboration on my graduate career.

Dr. Junkins, thank you for your help so far, and I hope to continue benefiting from your guidance during my PhD studies. Dr. Liu, thank you for agreeing to serve on my committee, and for sharing your thoughts which were quite helpful in both detailed and general ways. Ms. Karen Knabe, thank you for helping me to enroll, to achieve my first "warm-up" graduation, and I expect in just a few more brief years, to get to the big one. Ms. Colleen Leatherman, thank you for helping me do all of the things I need to do in between those times.

Finally, I would thank Mr. L. Kneip, who made his P3P solver code freely available online, as this was of great assistance to me both in benchmarking his algorithm, and subsequently modifying it. I am grateful to him for his act of academic cooperation.

This thesis has been an excellent learning experience, and I thank all those who helped me and I look forward to more to come.

NOMENCLATURE

6-DOF	6 Degrees of Freedom
ARD	Autonomous Rendezvous and Docking
CRP's	Classical Rodriguez Parameters
EKF	Extended Kalman Filter
GLSDC	Gaussian Least Squares Differential Correction
OLAE	Optimal Linear Attitude Estimator
P3P	Perspective-3-Point
PNP	Perspective-N-Point
PCC	Point Cloud Correspondence
SVD	Singular Value Decomposition

TABLE OF CONTENTS

Page

LIST OF FIGURES

Page

LIST OF TABLES

Page

CHAPTER I

INTRODUCTION

This thesis is motivated by the problem of developing a relative navigation system for small satellites, suitable for autonomous rendezvous and docking (ARD). We consider a sensing system based on measurement of the direction from a sensor housed onboard one spacecraft, to each of several geometric reference points on the structure of a second spacecraft, commonly referred to as vector measurements. To use such a sensor system, an algorithm must be found to solve for spacecraft relative position and orientation, known as pose, using these vector measurements as inputs. This scenario is illustrated in Fig. 1.

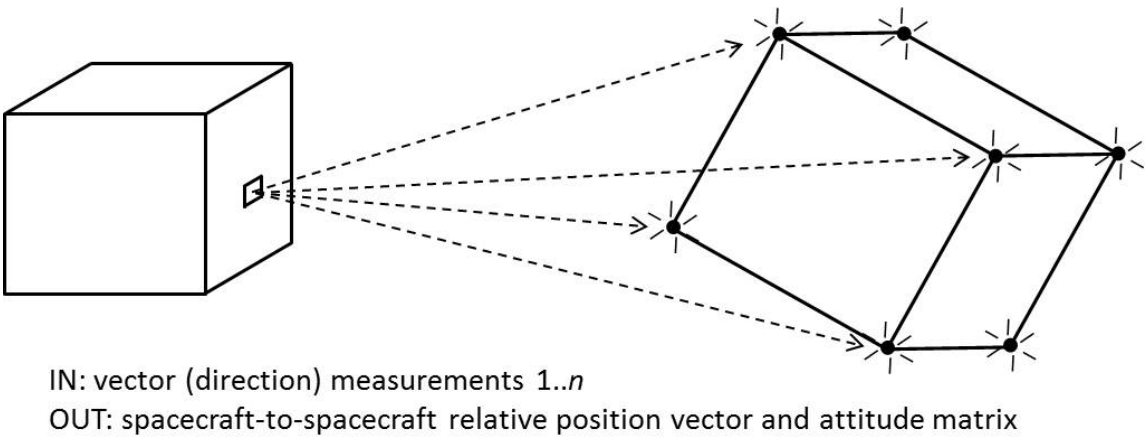

Fig. 1 Illustration of spacecraft relative navigation based on vector measurements.

This thesis follows the style of the *Journal of Guidance, Control and Dynamics*.

A number of means of solving this problem are available. The goal of this study is to arrive at a solution technique that reduces processing burden, given the constrained computational environment onboard a small satellite.

The problem of pose determination using vector measurements is mathematically equivalent to the Perspective-N-Points (PNP) problem, which is important in the field of computer vision today, and historically was first considered in the field of photogrammetry. The PNP problem was concisely stated by Fischler and Boelles in 1981 [1] as follows (with the parameter *m* in place of *N*):

> *"Given a set of m control points, whose 3-dimensional coordinates are known in some coordinate frame, and given an image in which some subset of the m control points is visible, determine the location (relative to the coordinate system of the control points) from which the image was obtained."*

A main focus of this thesis relates to the Perspective-3-Point (P3P) problem, which is an important subclass of the PNP problem where N=3 control points are known. An illustration of the P3P problem, placed in a historical context of aerial photogrammetry, is shown in Fig. 2.

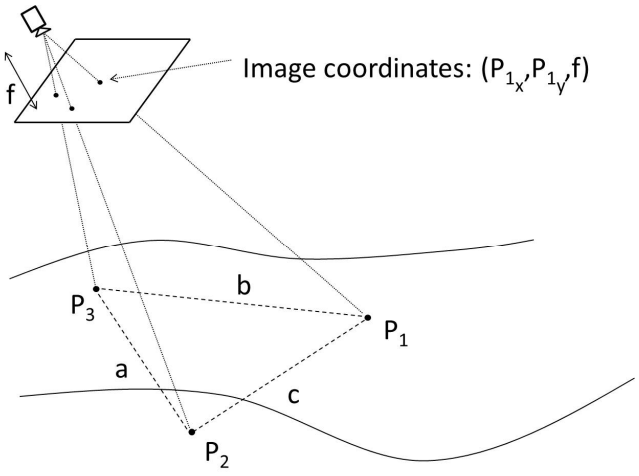

Fig. 2 Illustration of the P3P problem.

This problem is often considered with regard to three different types of measurement applicable to a camera's reading of a point in space, and they are all equivalent. These are: image plane measurements (coordinates measured on the image focal plane), unit vector measurements, and angle measurements. These three measurement types are illustrated for a spatial point with location given by [P_x, P_y, P_z] in Fig. 3. In this discussion, we will refer to whichever of these measurement types is most convenient for a given topic.

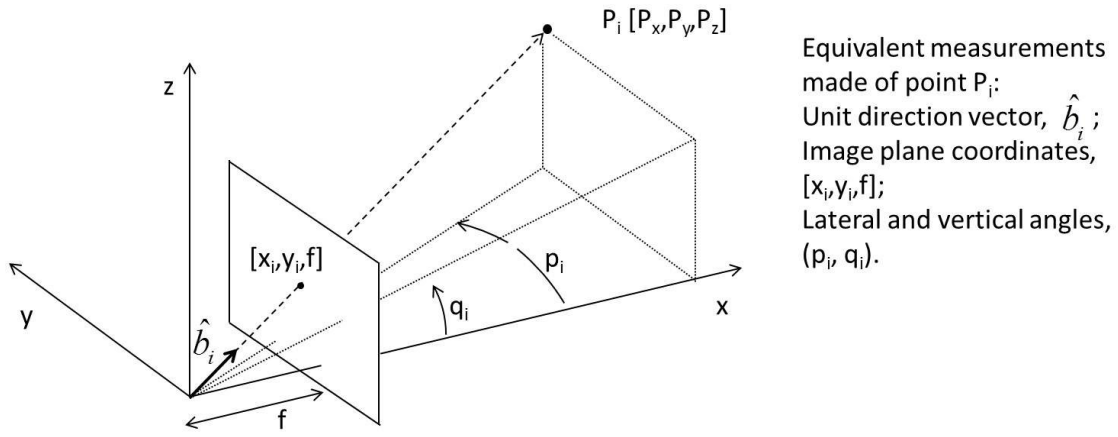

Fig. 3 Equivalent PNP measurements: image plane coordinates, lateral and vertical angles, unit vectors.

We began by considering a target spacecraft (referred to generally as the target) whose shape is a square or rectangular extrusion. The spacecraft is equipped with optical emitters suitable for a direction-sensing sensor. We assume hereafter that the emitters, which constitute the control features of the PNP problem, are located at the vertices of the cube. This special case was motivated by consideration of cubesats, which are cube-shaped small satellites. In particular, a "cubesat" standard has been created, by California Polytechnic State University and Stanford University [2]. This standard was designed to support a class of spacecraft that is low-cost and relatively simple in its design, while enabling the demonstration of new spacecraft technologies.

Cubesats can come in a range of sizes. The most common are '1U' (approximately 10x10x14 cm^3 bus size) and '3U' (approximately 10x10x34 cm^3). Larger sizes are also

available. An orthogonal (cube, or rectangular extrusion) geometry is adhered to in all cases.

Many other spacecraft that are not built to the cubesat standard, particularly larger university-class spacecraft, still often have orthogonal geometry. To illustrate real spacecraft that might be relevant to the relative navigation studies performed here, several photographs of cubesats and other university-class satellites with orthogonal geometries are shown in Fig. 4.

Upon seeing that most P3P problems were solved for control points in a general configuration, while in the cubesat relative navigation problem we expected to work only with orthogonal geometries, we were motivated to ask if a simpler solution tailored to this geometry could be determined.

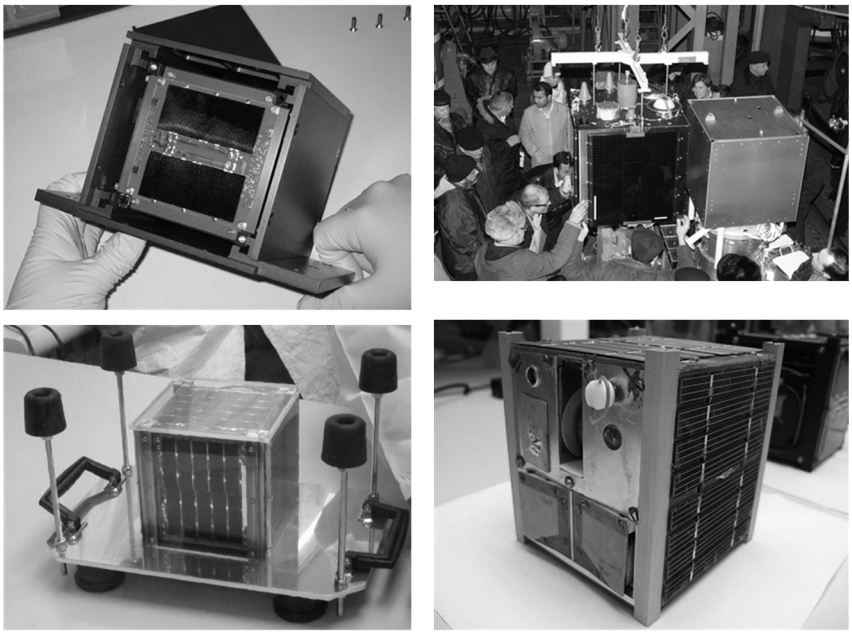

Fig. 4 Cubesats and other university satellites with orthogonal geometry [3].

To this end, we examined three P3P solution algorithms. The first was an original solution developed by this author, which was developed for the orthogonal geometry case from the outset. The second was a method developed in 1991 by Linnainmaa et al [4]. The third was a method developed by Kneip et al. in 2011 [5]. Of these, the Kneip method was found to offer the best performance for the general problem, and also to be well suited for tailoring for an orthogonal geometry. Although P3P solutions abound, based on the recent publication date and on the findings presented in [5], as well as our own findings that are presented here, it seems reasonable to assume that the Kneip solver can be considered a state-of-the-art P3P solver.

By simplifying the mathematics of the Kneip solution to reflect the known perpendicular geometry, we achieve a reduction in computation time of the P3P solution of over 40%. To our knowledge, this question, of using knowledge of the shape of the target to reduce the computational load of the P3P problem, has not been examined previously. This is clearly due in part to the fact that in many PNP applications, nothing less than general target geometries can be considered. However, we mention here two cases in which a solver might feasibly be limited to a geometry of perpendicular features.

The first is the case in which the target is designed as part of the system. The cooperative spacecraft relative navigation problem that motivated this study is an example. While this idea was motivated by the perpendicular geometries found on cubesats, a perpendicular arrangement of target points could be created without requiring that the underlying structure be perpendicular. Thus, a perpendicular arrangement of

target points is expected to be possible in any case where the target is designed as part of the system.

Fig. 5 Photograph of man-made environment (research lab), with orthogonal features available for navigation.

The second case is where the target is not designed as part of the system, but nevertheless features orthogonal points: specifically, man-made structures. Fig. 5 shows a scene where, from inspection, it would appear that a robot using orthogonal targets for

navigation may well have enough “naturally” occurring targets to navigate the environment successfully.

Solving the P3P problem does not generally yield a unique solution; rather, up to four distinct solutions will be possible. Fig. 6 and Fig. 7 show illustrations of the two-solution and four- solution cases. At least one additional control point must be used to disambiguate the correct solution. If measurements are perfect, this will unambiguously result in the correct solution. If measurements include error, no correct solution is obtainable, and an optimization method must be used.

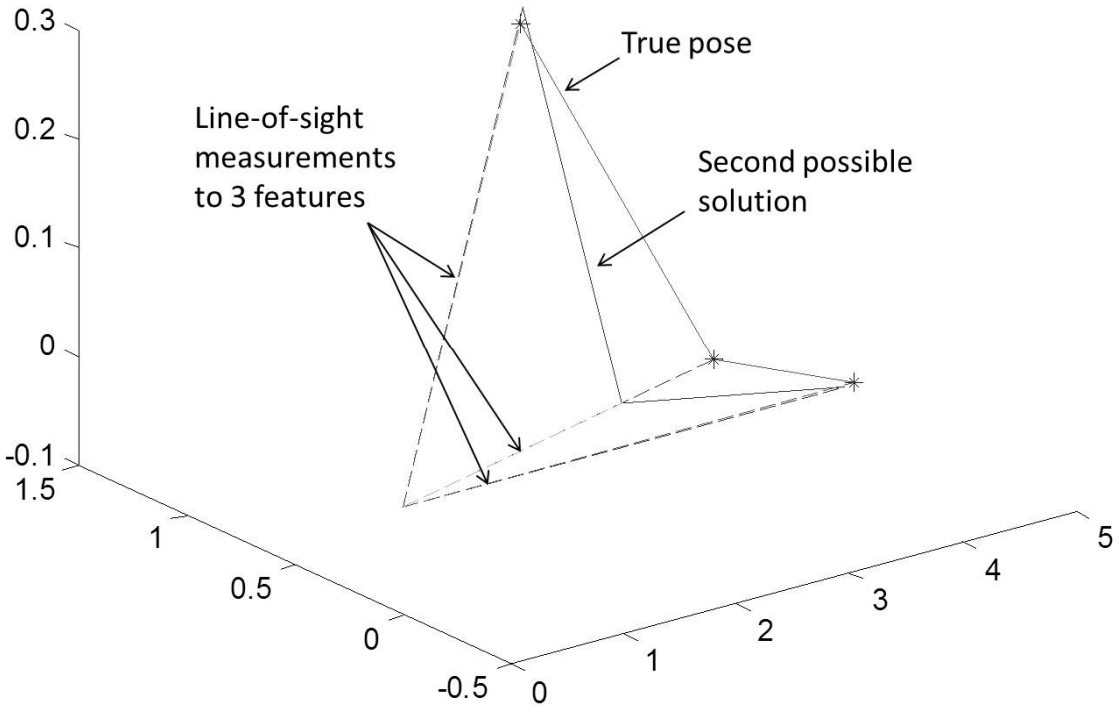

Fig. 6 Plotted example of a P3P problem with two ambiguous solutions.

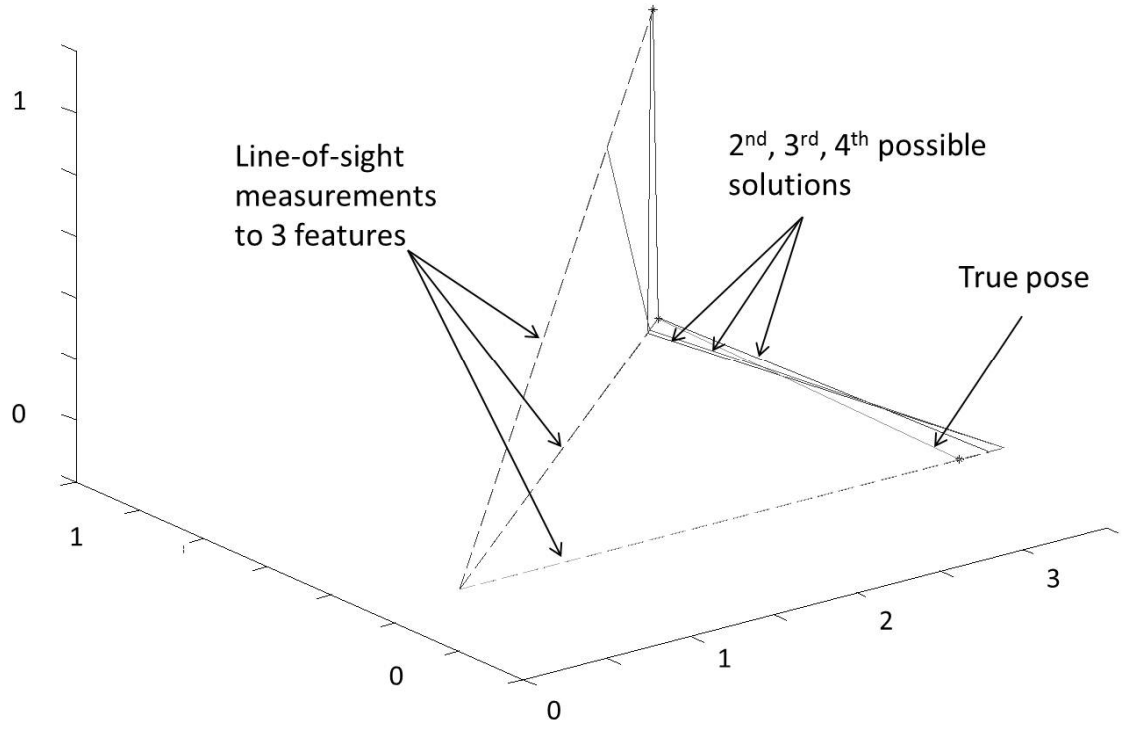

Fig. 7 Plotted example of a P3P problem with four ambiguous solutions.

In this thesis, we chose to follow the method employed by Kneip, and thus disambiguated using a point cloud correspondence (PCC) technique. However, whereas Kneip solves the PCC problem using Arun's method [6], we employ the Optimal Linear Attitude Estimator (OLAE) method, as described in [7] and [8], which is expected to perform a similar (but not identical) optimization at reduced computational burden. To complete the line of questioning of this thesis, we also tailor the geometry of the OLAE solver, and examine the effects on computational time. Here, it was found that tailoring could equally well be performed with any geometry, and the reduction in computation

time is only around 10%; and it seems from examining the computational steps involved that this solver does not offer as many opportunities for simplification via this approach. However, the processing burden of the P3P solver was an order of magnitude larger than that of the PCC/OLAE solver, so the improvement to its efficiency has a larger impact on total computation time.

With these two steps, we find the solution of a P4P problem given real-world measurements, sufficient to find a pose solution from vector (line-of-sight, angle, image plane, etc.) measurements. Our findings related to optimizing the solver for target geometry are specific to the solution approach and the specific solvers chosen in this study. However, we believe the findings suggest the possibility that some level of improvement can be achieved in any PNP solution implementation, if target geometry can be known in advance.

CHAPTER II

LITERATURE REVIEW

We divide the literature review for this problem into in three parts. First, we review past publications on the topic of performing spacecraft relative navigation using vector measurements. This problem is the motivation for this study, and since it has been examined by others within the aerospace engineering community, we begin with those efforts. Second, we discuss literature related to the P3P and PNP problems. Third, we highlight previous P3P and PNP efforts which have examined specific target geometries, and compare the direction taken in those works, to that taken here.

A. Spacecraft relative navigation using vector measurements

A number of researchers have examined the capabilities of a spacecraft relative navigation system based on vector measurements. Closest to home, Maeland et al. in 2010-2011, at the AggieSat Lab at Texas A&M University, considered implementation of such a system based on using sensors found in a consumer-grade product, specifically a programmable robotic ground vehicle [9]-[12]. The sensor, which provides measurements for up to five emitters simultaneously at distinct optical frequencies, is shown in Fig. 8. This approach was motivated by the low cost of the sensor, compared with similar space-rated sensors which were generally an order of magnitude larger, more costly and more complex. Attention was given to experimental characterization and calibration of the sensor, and understanding the sensor's internal signal conditioning.

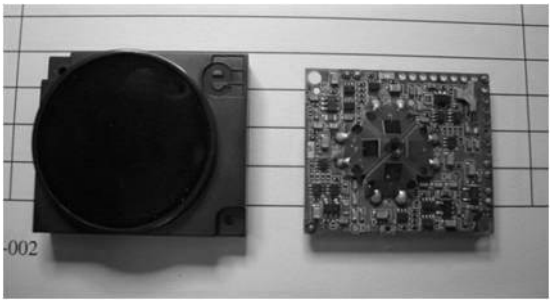

Fig. 8 NorthStar optical sensor from Evolution Robotics [13], considered as a RelNav sensor for small satellites in [9].

These works developed alongside other efforts at Texas A&M University. Du in 2004 considered a different sensor with its own distinct properties [14],[15]. This study considered Extended Kalman Filter (EKF) as well as GLSDC methods for finding the pose solution. With consideration of this same sensor, Crassidis et al. in 2001 presented a new pose solution algorithm derived from a generalized predictive filter for nonlinear systems was presented [16]. Mortari et al. in 2004 presented three new solutions, of which two require iteration, while the third is a direct solution based on linear algebra [17]. This third solution requires a minimum of six control features to arrive at a unique solution, which is in agreement with other findings (e.g. those of Fischler and Boelles in [1]) that at least six control features in general position are required, although four control features are sufficient if they are coplanar.

Other researchers have also considered this problem. Mukundan and Ramakrishnan in 1995 parameterized spacecraft attitude using quaternions, and then found a direct

solution for relative pose [18]. However, the solution involved an approximation, which is not inherent to relative pose solutions in general.

In another study, Woffinden and Geller in 2006 examined a guidance and control system relying heavily on line-of-sight measurements as inputs to a navigation filter [19]. Performance during various phases of a rendezvous mission was considered.

We will now review P3P and PNP solutions, which are expected to be applicable to the problem of spacecraft relative navigation using vector measurements.

B. The P3P and PNP problems

The P3P problem is an important instance of the PNP problem. Where N control features are considered, $N=3$ is the minimum value that will allow a finite number of solutions. This number can be 1, 2, 3 or 4. (Gao et al, 2003 [20]) Problems with $N=4$ and $N=5$ offer a unique solution in some but not all cases, and in all cases for $N\geq 6$. Problems where $N>6$ would be redundant given perfect measurements, but are important in the context of practical applications where measurements are not perfect, and the task becomes not to identify the "correct" solution (it cannot be identified given imperfect measurements), but rather to identify a best estimate by considering all of the measurements available. Although some avenues of research in the PNP problem are motivated by computer vision applications, and take up challenges not expected in the spacecraft relative navigation problem, the basic problem of arriving at a best-estimate PNP solution given imperfect measurements, with $N \geq 4$, is applicable to the spacecraft relative navigation problem.

1. Direct solutions to the P3P problem

The P3P problem was first encountered when performing scene analysis from photographs, in the field of photogrammetry. The problem first came to be known as the three point space resection problem. The first direct solution is considered to have been presented in 1841. (Haralick et al, 1991 [21]) Additional direct solutions were identified subsequently, some with and some without reference to prior solutions. The problem received significant attention in aerial photography, and later, computer vision.

Most direct solutions to the P3P problem are based upon the three-equation system (1) below, where (s_1, s_2, s_3) are unknowns, (a, b, c) are known geometry parameters, and (α,β,γ), the interior angles between two direction vectors, are measured, or derived from measurements. The equations are each applications of the Law of Cosines to one of the three triangles formed by the camera and two of the control features. The reference configuration, similar to Fig. 2 but with the variables of equations (1) labeled, is shown in Fig. 9. Direct solutions to the P3P problem entail solutions to equations (1).

$$\begin{aligned} s_2^{\,2} + s_3^{\,2} - 2s_2 s_3 \cos\alpha &= a^2 \\ s_1^{\,2} + s_2^{\,2} - 2s_1 s_2 \cos\gamma &= c^2 \\ s_3^{\,2} + s_1^{\,2} - 2s_3 s_1 \cos\beta &= b^2 \end{aligned} \tag{1}$$

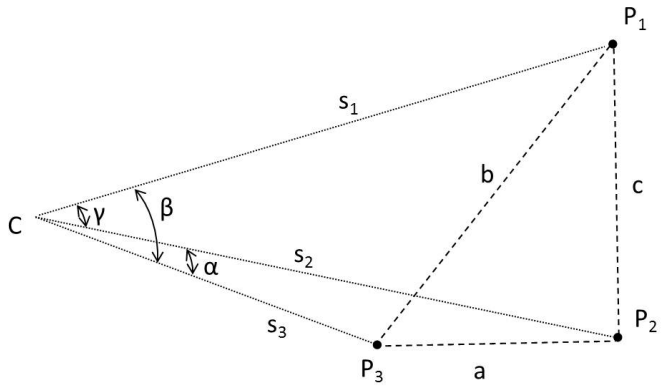

Fig. 9 P3P geometry with variables labeled.

As posed above, this problem has three equations in three unknowns. However, formally there are six unknowns: three for relative location, and three for relative orientation. The formulation above can be considered to collapse these six down to three, by solving for three scalar lengths, given that these scalars result in vectors between the camera and each target point, when multiplied by the unit vectors derived from measurements that point to each feature. It is by this step that the solution of three scalar values is then taken to determine a full 6-degree-of-freedom (6-DOF) solution.

A large number of solutions to this problem can be found in the literature. Generally, the motivation for considering new solutions is to increase numerical stability, and reduce computational time.

Haralick et al. provided as of 1991 a review of the most noted direct solutions [21]. One of those solutions, the Linnainmaa solution, was benchmarked in this study and will be discussed further in later sections. Below, we describe the typical solution process.

The simplest means of arriving at a solution is through algebraic substitution, to manipulate equations (1) into a single equation in a single variable. If done in the most straight-forward manner, this will be a fourth-order polynomial, or an eighth-order even polynomial. The fourth-order polynomial gives rise to four solutions, not all of which are real-valued. Solutions with large imaginary content can be discarded. The eighth-order even polynomial gives rise to four solutions in front of the camera, and four identical solutions behind the camera which can be discarded; hence it is an equivalent formulation. For a presentation of the math accompanying this description, [21] is recommended.

As reported in [21], there is another solution, by S. Finnsterwalder, that reduces the problem to a cubic coupled with two quadratic polynomials, instead of one quartic polynomial, which may have advantages in numerical stability. However, an English transcription of Finnsterwalder's work could not be found, so this solution was not pursued.

An interesting perspective on the problem was taken in works such as (Gao et al, 2003 [20]), where instead of asking, "Given a set of P3P measurements, how can we solve the problem to arrive at 1, 2, 3 or 4 solutions which can then be disambiguated using a fourth point," the question was changed to, "Given a set of P3P measurements, how can we anticipate whether 1, 2, 3 or 4 legitimate solutions will be returned?" This appeared to be an interesting (and complex) mathematical take on the problem. However, the value from an application standpoint is not clear, since if the legitimate

number of solutions is 2, 3 or 4, which it almost always is, the necessity of disambiguating using a fourth point remains.

The P3P problem can also be solved using iterative methods, such as GLSDC. If only the P3P problem must be solved, a direct solution will be computationally faster than an iterative one. However, if the P3P solution is occurring as part of a larger PNP problem, iterative schemes are often considered. This is discussed in the next section.

2. Relationship of the P3P problem to the PNP problem

Solving the P3P problem is typically done as the first step toward solving a PNP problem. Let us consider a PNP problem of $N \geq 6$, with no further specifications as of yet. We can pick any three of these points and solve the associated P3P problem. But what has been accomplished? We now consider some important distinctions in the problem based upon the type of measurements expected, that will decide our course. Please refer to Table 1.

Table 1 Breakdown of the PNP problem according to nature of measurements.

1. Measurements are perfect.	Of the ≤ 4 P3P solutions returned, one is precisely correct. Pick any fourth point to disambiguate.
2. Measurements include noise or small error.	None of the ≤ 4 P3P solutions returned are precisely correct. Consider all additional points to arrive at a best estimate.
3. Measurements include noise or small error, and some may be entirely wrong ("gross error").	P3P solutions from many measurement sets must be compared, to identify and discard the "gross error" measurements. From there, proceed from #2. See [1].

We present the PNP problem in this manner because a great deal of the literature is focused on the first (which amounts to the P3P solution) and third items. The third item is considered in computer vision, where (paraphrasing Fischler and Boelles in [1]) an automated system must identify the target's features in the image space before it can determine pose from the image coordinates. In other words, the computer must analyze a frame of a video image coming in from a video camera (for example), and identify the target features within the digitized content of this image – which sometimes happens erroneously. Computer vision experts have been motivated to develop PNP solver algorithms which tolerate not only noisy measurements, but also measurements that are entirely incorrect.

We expect that the spacecraft relative navigation from vector measurements problem can be placed in the second category. In the main conception, the system is designed on

both ends (sensor and target), and an active rather than passive target signature is used. A "gross error" measurement would have to come from another source of infrared emission similar to the emitters used, including frequency modulation. This is not expected as an environmental effect. It might arise in the context of a deliberate jamming activity, but we do not address that contingency here.

With this in mind, we will discuss PNP solutions in the literature, commenting briefly on the work addressing problems with gross error, but only examining closely the mechanisms required to address disambiguation, and noisy measurements.

3. Iterative PNP solutions

We draw again on [1], which, although it focuses on techniques to address gross error, also discusses a generic iterative approach. As described in Table 1, a PNP solution can be performed by first selecting three of the N points and finding the resulting 1, 2, 3 or 4 P3P solutions. Since the measurements are noisy, none of the solutions will in general agree perfectly with the other measurements, so disambiguation may not be clear cut. The objective then is only to find a starting guess for an iterative scheme. Employing an iterative scheme such as GLSDC to arrive at a converged estimate incorporating all the measurement sets, and taking a P3P solution as a starting guess, is one means of solving the PNP problem.

It is also possible to do away with the P3P solution, if another suitable starting guess can be found. With regard to spacecraft relative navigation, a direct P3P solution to as a starting guess could be used to achieve first acquisition. Once initial pose has been acquired, and the challenge becomes to make updates to spacecraft relative motion,

using the spacecraft's previous state as an initial guess to calculate its present state may be suitable.

4. Direct PNP solutions

Direct methods have also been developed for solving the PNP problem. Given perfect measurements, the simplest direct method is to solve the P3P problem with three points, finding up to four possible solutions, and then pick any fourth point to disambiguate the correct solution. However, given measurements with error, the problem becomes one of optimization. Many direct solutions have been considered; we mention a few here.

One approach given measurements with error is to proceed with a P3P solution and disambiguation to achieving an incorrect and non-optimal solution, and then further improve the resultant pose solution. This can be done by treating the disambiguated pose solution vectors as "measurements" provided as inputs to a problem we term the point cloud correspondence (PCC) problem (also called fitting of 3-D point sets, and probably other names as well). This problem seeks to find best estimated values for a rotation matrix R and a translation vector T given measurements p_i' with noise N_i of body vectors p_i, produced by:

$$p_i' = Rp_i + T + N_i$$

An illustration of this problem is shown in Fig. 10.

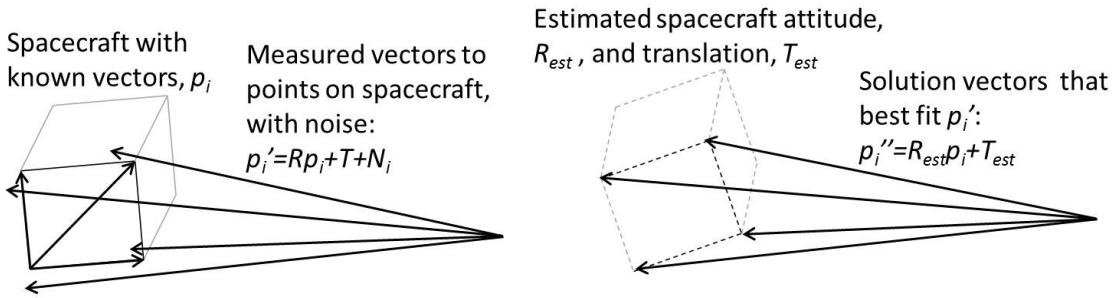

Fig. 10 Illustration of PCC problem.

As will be discussed later, a PCC solution is not expected to be an optimal estimation for the PNP problem, because it does not directly optimize the error from the original vector (angle, line-of-sight, etc.) measurements. But given small noise levels, the solution may be appropriate; and it can be done with little computation time.

Arun et al. in 1987 gives one solution to the PCC problem [6]. Mortari et al. in 2007 give another solution, known as OLAE [8]. OLAE is presented as an attitude estimator, but can be applied to PCC because translation is decoupled in this problem (for discussion of this, see [6]).

Other direct PNP solutions do not begin with a P3P solution. For example, the solutions presented in (Haralick, 1989 [22]) and (Penna, 1991 [23]), mentioned later, amount to direct P4P solutions. The linear algebra solution in [17] amounts to a direct P6P solution, and is both rigorous and optimal given small measurement errors. And Hesch in 2011 provided a recent direct solution given an arbitrary number of points [24]. This solution is also rigorous, and optimal given measurements with error. The solution

is found by minimizing a cost function that is quartic in the state parameters. We will mention this solution again later, in the context of future work.

In this study, the PNP problem is solved by finding the P3P solution, disambiguating with a fourth point, and then improving the solution using PCC techniques, specifically those of Mortari et al. in [8]. We chose this route because PCC techniques were also used by Kneip.

We summarize this section with the flow chart presented in Fig. 11, which shows various distinct approaches to solving the PNP problem along the lines discussed here. The techniques that are addressed in this study, the P3P direct solution and direct determination of a disambiguated solution accounting for noise, are indicated in the flowchart with bold outline.

C. Previous P3P and PNP examinations that considered target geometry

This section presents previous studies of the P3P and PNP problems that examined specific rather than general target geometries. Although the entire body of work in this problem is large and widely scattered and could not be sampled entirely, this presentation is based on a methodical literature search that, we expect, has revealed the bulk of relevant past work. The goal of this section, therefore, is to support a conclusion that the question of achieving computational advantages in P3P or PNP solutions by taking advantage of known target geometry has, at minimum, not been addressed previously with a controlled study.

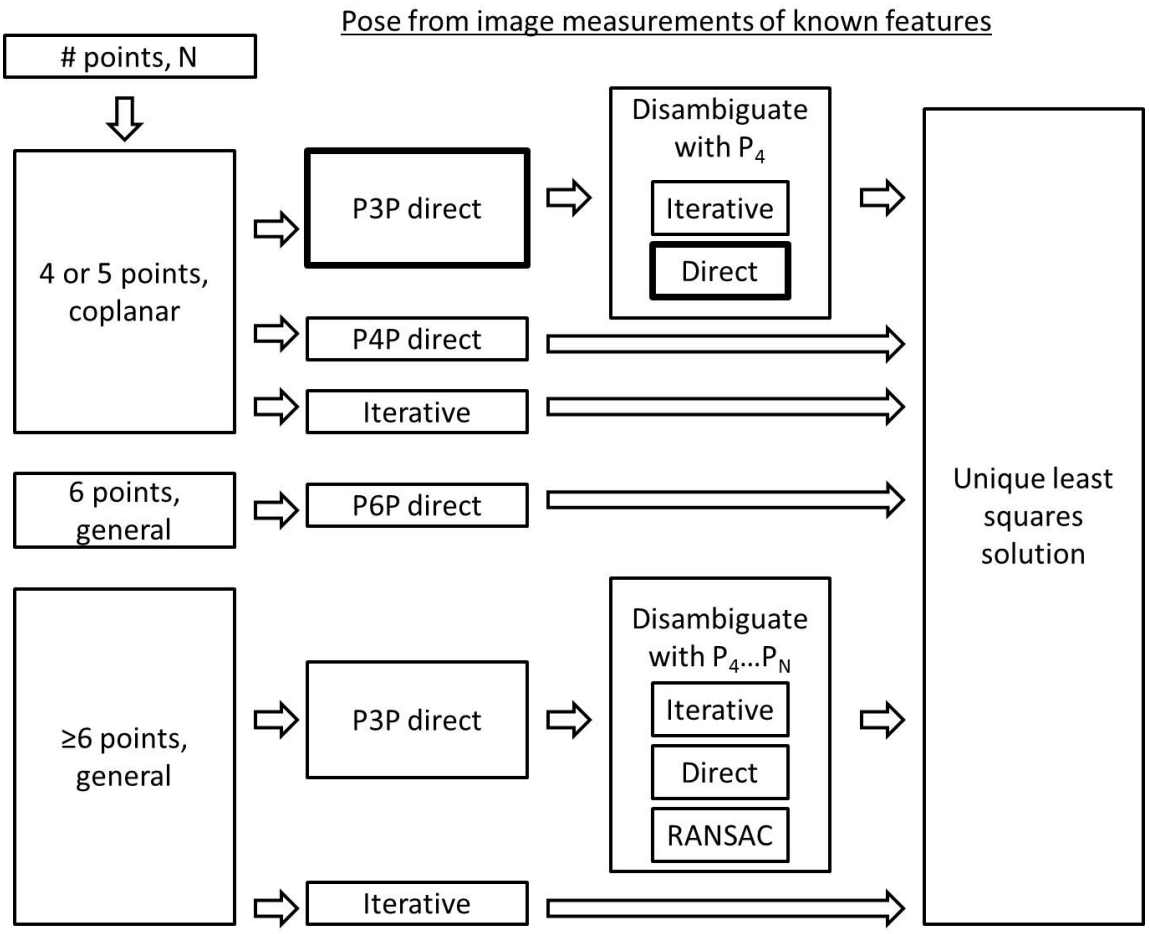

Fig. 11 Flow chart of approaches to solving the PNP relative pose problem.

Our literature survey begins with two past studies that lie closest to this work. Haralick in 1989 considered a target in the shape of a rectangle [22]. However, the objective was not to reduce computation time, but rather, to prove mathematically that the attitude portion of the pose problem can be solved using only a target (four feature points) that is known to be a rectangle.

Lee et al. in 1990 also consider orthogonal targets, in this case focusing on a robotic vision location method in man-made environments [25]. The potential for using rectangular object references to this purpose is mentioned here. The author examines computational performance, but the focus is on error. The author does emphasize that simple equations can be formulated in his method. However, he makes this statement without reference to what the equations would have looked like had the target been a different geometry. The first motivation stated for choosing the rectangle geometry was to provide clear features for a feature detection solver.

Penna in 1991 extended the results of [22], the P4P solution for a rectangle, to general quadrilaterals [23]. These two papers can be seen as setting up a comparison between computational performance for rectangular vs. general quadrilateral targets which would be analogous to our study, but for a P4P rather than P3P solver. However, to our knowledge this comparison has not been performed.

Mukundan and Ramakrishnan, 1995 [18] took a specific target geometry for their analysis, shown in Fig. 12. However, the geometry appeared to be chosen for general engineering factors, and the impact of the choice of geometry on the problem was not considered. (It is not clear how the author avoided the multiple-solution ambiguity with a target consisting of only three points. Possibly, small motions were used between increments, so that continuity with previous solutions could be used.)

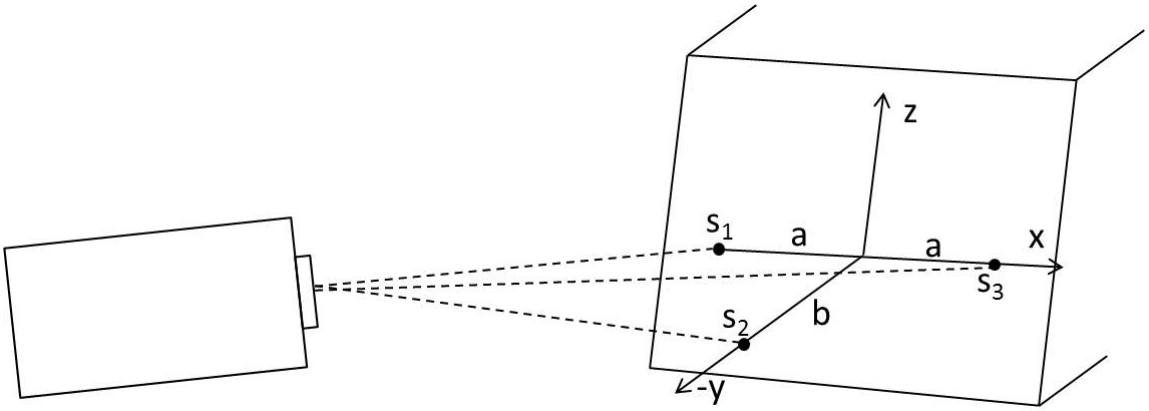

Fig. 12 Target geometry considered by Mukundan for spacecraft RelNav [18].

Tang et al. in 2008 considered an orthogonal target geometry, as used in this study, was considered [26]. The author also mentions the interest in orthogonal targets due to their occurrence in man-made structures. However, this paper concerns itself, as did Gao et al. in [20], with performing solution classifications (i.e. determining based on input measurements whether a P3P problem will return 1, 2, 3 or 4 solutions). Metrics related to practical implementation, such as computational load vs. non-orthogonal targets, were not considered.

CHAPTER III

COMPARISON OF BENCHMARKED SOLUTIONS

In this study, a general P3P solver was selected for tailoring to orthogonal targets. However, prior to this step, three different P3P solvers were implemented and compared. One of these solvers was developed originally in this work, and worked only for orthogonal targets. It was compared against two pre-existing solvers that handled targets of general geometry. Had the comparison demonstrated the superiority of the original solver developed here, in terms of speed and numerical stability, no further modification would have occurred. However, one of the other solvers proved superior, even at solving for orthogonal targets, to the solution developed here for this purpose, and so this solver was selected and subsequently tailored for orthogonal targets. In this section we discuss the comparison of the three benchmarked solvers.

A. General discussion of the three benchmarked solvers

The three solvers selected for benchmarking were: the “Linnainmaa solver,” by S. Linnainmaa et al, described in [4]; an original solver, described below; and the “Kneip solver,” by L. Kneip et al, described in [5], and further here in Chapter 3. Each solution is described below.

1. Linnainmaa solver

The Linnainmaa solver was selected because it appeared to be the most recent solution named in the review of the P3P problem presented in by Haralick et al. in [21], and because a description of it was available in the literature. For brevity, we will not present its solution mechanism here, other than to say that it represents a solution to equations (1) via algebraic manipulation, resulting in a 4^{th} order polynomial, as is typical to P3P solutions.

2. Original solver

The original solution developed here worked somewhat differently, for two reasons. The first was that it was restricted in its system of equations to orthogonal targets. (These equations could be modified for general targets, but not easily.) The second is that this solver began by performing a coordinate system transformation to a new coordinate system oriented relative to the target, definable using the camera's measurements of the target. This solution was thus explicitly a three-step process: 1) change coordinate systems; 2) solve the problem in the coordinate system oriented on the target; 3) convert the answer to the general camera coordinate system.

The coordinate system transformation used by this solver happens, coincidentally, to be almost exactly that used by the Kneip solver. The two differ only by choice of axes. We feel it is helpful to clarify that we developed this method in May-July of 2011. Kneip's paper [5] was published in August 2011. Our purpose here is not to take from Kneip's work, but only to present the work that we did as independent.

The coordinate system transformation consists of three rotations: a yaw and a pitch, such that the rotated x-axis points directly at the first target point (P_1); and then a roll, such that the half-plane formed by the x-axis and the positive z-axis contains the second target point (P_2). The angles of the three rotations are depicted in Fig. 13. Of the three target points, P_3 is not shown because the coordinate system transformation is independent of P_3.

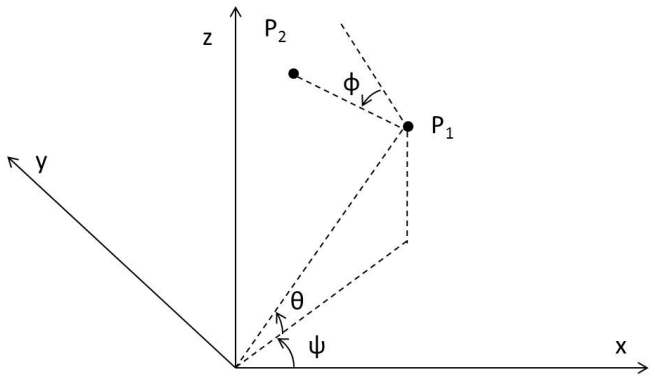

Fig. 13 Diagram of three angles (ψ, θ, φ) used to perform 3-2-1 rotation from generic camera coordinate system to system oriented on target.

The angles (ψ,θ,φ) must be identified from measurements. We recall that measurements can be considered equivalently as unit vectors, or as angle measurements. Considering angle measurements (p_i,q_i), where p_i is the angle to the projection of point P_i in the x-y plane and q_i is the same for the x-z plane (as illustrated in Fig. 3), (ψ,θ,φ)

can be obtained by performing the rotations one by one and finding transformed measurements at each step, as follows:

$$\psi = p_1$$

$$\theta = q_1'$$

$$\tan\varphi = \frac{\tan p_2''}{\tan q_2''}$$

where q_1' represents q_1 transformed according to the ψ (yaw) rotation, and p_2'' and q_2'' represent these angles after both the ψ (yaw) and θ (pitch) rotations.

The next task is to transform the measurements into the new coordinate system. This can be done in one of a few ways. If the measurements are considered as unit vectors, then they can be transformed immediately with the use of a rotation matrix for an Euler 3-2-1 rotation given by (ψ,θ,φ) above. If the measurements are angle measurements, then they can be transformed as such by using careful geometry analysis. Classical photogrammetry methods are relevant, such as those presented in [27]. This was the first approach we took. Alternatively, "virtual points" can be constructed from the angle measurements, and these can be transformed with the Euler rotations method, and then converted back into angles. Finally, regarding unit vector measurements, Kneip shows another method, which we will see later, that is simple and elegant.

Now we discuss the effect of the coordinate system transformation. Prior to the transformation, the coordinates of any target point P_i are:

$$[\mathrm{P_i}]_C = [x_i, y_i, z_i]^T$$

In other words, they are generally placed. We are considering for the moment a general rather than orthogonal target. After the coordinate system transformation, the coordinates are now:

$$[P_1]_C = [\ r\ 0\ 0\ ]^T$$

$$[P_2]_C = [\ a\ b\ 0\ ]^T$$

$$[P_3]_C = [\ c\ d\ e\ ]^T;$$

where the non-zero values would be unknowns. The three coordinates that have been reduced to zero represent three degrees of freedom that have been removed from the problem by the three rotations (ψ,θ,φ). Also, r above is a scalar range from the camera to the first point.

We will now parameterize this representation considering an orthogonal (rectangular) target. Points P_1, P_2, and P_3 describe two adjacent sides of the target. Take a normalized unit of length equal to the side (P_1, P_2). The range solution will be in terms of this normalized length. Then take a multiplier, u, denoting the scaled length of side (P_2, P_3), where $u=1$ would denote equal side lengths, i.e. a square.

Now we define two orientation parameters:

α: the angle between the side (P_1, P_2) and the x-axis; and

β: the angle between side (P_2, P_3) and the x-z plane.

The geometry and orientation parameters are shown in Fig. 14. We have now taken distance units normalized to edge (P_1, P_2), such that the length of that edge is 1, and the length of edge (P_2, P_3) is u, which relates the aspect ratio of the rectangular shape: i.e. $u=1$ would correspond to a square target.

The coordinates of the three points are now as follows:

$[P_1]_{C'} = [\ r \qquad 0 \qquad 0\ ]^T;$

$[P_2]_{C'} = [\ r+cos(\alpha) \qquad sin(\alpha) \qquad 0\ ]^T;$

$[P_3]_{C'} = [\ r+cos(\alpha)-u\ sin(\alpha)cos(\beta) \qquad sin(\alpha)+u\ cos(\alpha)cos(\beta) \qquad u\ sin(\beta)\ ]^T;$

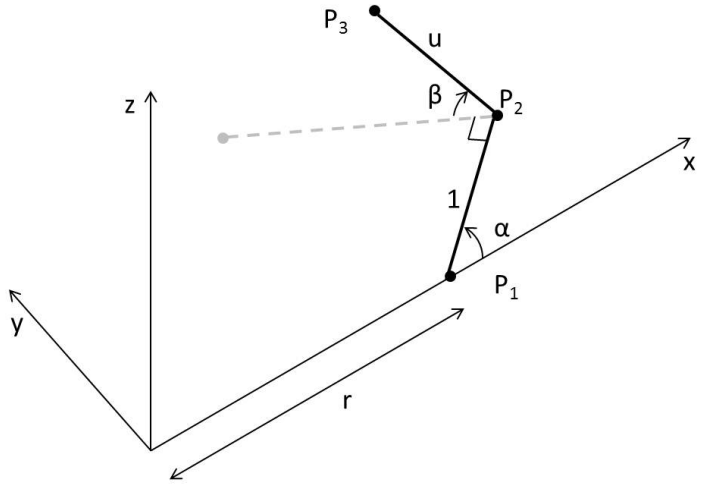

Fig. 14 Geometry of pose determination problem for orthogonal target in transformed coordinate system, with remaining state variables (*r,α,β*).

We can now define the system equations, i.e. the equations that relate state variables to measurements. In this case, we consider angle measurements (p_i, q_i), where, with perfect measurements, $\tan p_i = \frac{[P_i]_y}{[P_i]_x}$ and $\tan q_i = \frac{[P_i]_z}{[P_i]_x}$. This leads to the equation system:

$$\begin{aligned}\tan q_2 &= d = \frac{\sin\alpha}{r+\cos\alpha} \\ \tan p_3 &= e = \frac{u\sin\beta}{r+\cos\alpha-u\sin\alpha\cos\beta} \\ \tan q_3 &= f = \frac{\sin\alpha+u\cos\alpha\cos\beta}{r+\cos\alpha-u\sin\alpha\cos\beta}\end{aligned} \tag{2}$$

Analogous to equations (1), a solution to equations (2) in (r,α,β) allows a solution to the P3P problem. The solution must then be translated back into the camera coordinate system. In our study, this system was solved algebraically using substitutions ($\sin\vartheta \to p, \cos\vartheta \to \sqrt{1-p^2}$), to arrive at an 8th order even polynomial, and thus at a 4th order polynomial, as is typical of P3P solutions. Because sign information on both α and β is lost in the algebra, a quadrant check is performed in post-processing of the solution.

3. Kneip solver

The Kneip solver has conceptual similarities with the original method described above. However, Kneip sets up his system in a way that allows for extremely clean and simple computations.

Kneip sets out to solve for camera pose in a world frame in which the target is placed, rather than in the target frame directly. This consideration is not essential to the problem of spacecraft relative navigation, but we maintained the structure in Kneip's formulation while defining the frames as equivalent: $[\vec{v}]_w = [\vec{v}]_\eta$, for all $\vec{v}$.

In the next section, we show how the Kneip solver was modified to create a solver tailored specifically for orthogonal targets, and thus offer even more gain in computation

time. Here we present the Kneip solver in adequate detail to support that subsequent discussion, with the understanding that we are replicating the presentation of [5].

Kneip takes unit vectors $[\vec{f}_1, \vec{f}_2, \vec{f}_3]$ pointing to target features as his input measurements. He then performs a coordinate system transformation that is similar to the one used in our original method. Two differences are noted.

First, Kneip aligns the positive *x*-*y* plane, rather than the *x*-*z* plane, with edge (P_1,P_2). This difference is semantic only.

Second, Kneip defines his coordinate system transformation elegantly. He constructs the matrix *T* that transforms vector coordinates from the camera ($_c$) frame to the target-pointing ($_\tau$) frame in terms of his measurements, as:

$$\begin{aligned}
\vec{t}_x &= \vec{f}_1 \\
\vec{t}_z &= \frac{\vec{f}_1 \times \vec{f}_2}{\left|\vec{f}_1 \times \vec{f}_2\right|} \\
\vec{t}_y &= \vec{t}_z \times \vec{t}_x \\
T &= \left[\left[\vec{t}_x\right]_c \left[\vec{t}_x\right]_c \left[\vec{t}_x\right]_c\right]^T
\end{aligned}$$

Similarly, Kneip uses his feature point vectors, expressed in the world frame, to create a transformation matrix *N* from the world frame ($_w$)to the target frame ($_\eta$):

$$\begin{aligned}
\vec{n}_x &= \frac{\left[\vec{P}_2 - \vec{P}_1\right]_w}{\left|\left[\vec{P}_2 - \vec{P}_1\right]_w\right|} \\
\vec{n}_z &= \frac{[\vec{n}_x \times (\vec{P}_3 - \vec{P}_1)]_w}{\left|[\vec{n}_x \times (\vec{P}_3 - \vec{P}_1)]_w\right|} \\
\vec{n}_y &= \vec{n}_z \times \vec{n}_x \\
N &= \left[\left[\vec{n}_x\right]_w \left[\vec{n}_y\right]_w \left[\vec{n}_z\right]_w\right]^T
\end{aligned} \tag{3}$$

The target frame is considered centered on P_1, with its x-axis pointing to P_2, and its y-axis such that the positive x-y plane contains edge (P_1,P_3).

In this frame, the target geometry coordinates are given in parameters (d_{12},p_1,p_2) as:

$$[P_1]_{\eta'} = [\ 0 \quad 0 \quad 0\ ]^T;$$

$$[P_2]_{\eta'} = [\ d_{12} \quad 0 \quad 0\ ]^T;$$

$$[P_3]_{\eta'} = [\ p_1 \quad p_2 \quad 0\ ]^T;$$

Kneip then seeks to find the transformation matrix between the η and τ frames. So far, he has arrived at the same 3-DOF problem as was encountered with the original solver. Kneip's three remaining state space variables are:

$|CP_1|$: equivalent to r, the scalar distance from the camera (C) to point P_1.

α: equivalent to π-α from the previous method. The symbol α was used in both cases.

θ: equivalent to β.

Kneip's system of equations, in which he takes unit vector, $\hat{f}_i$, as measurements, is:

$$\begin{aligned}
&\frac{|CP_1|}{d_{12}} = \frac{\sin(\pi-\alpha-\beta)}{\sin\beta} \\
&\frac{[f_3]_\tau^T \cdot [100]^T}{[f_3]_\tau^T \cdot [001]^T} = \frac{-p_1\cos\alpha - p_2\sin\alpha\cos\theta + d_{12}(b\sin\alpha+\cos\alpha)}{-p_2\sin\theta} \\
&\frac{[f_3]_\tau^T \cdot [010]^T}{[f_3]_\tau^T \cdot [001]^T} = \frac{p_1\sin\alpha - p_2\cos\alpha\cos\theta}{-p_2\sin\theta}
\end{aligned} \tag{4}$$

with $\cos\beta = \vec{f}_1 \cdot \vec{f}_2$ and $b = \cot\beta$.

The unknowns are ($|CP_1|$,α,θ). Geometry parameters are (d_{12}, p_1, p_2). (In our system, d_{12} is normalized to 1.) The measurements are the two degrees of freedom of the unit

vector f_3, and the scalar $b(f_1, f_2)$. The first equation relates $|CP_1|$ and α to point P_2, and the second two relate α and θ to the lateral and vertical components of point P_3. These relationships mirror the relationships of the corresponding parameters in equations (2).

From here, Kneip uses algebraic substitution to arrive at a 4^{th} order polynomial in $cos(\theta)$. He solves this polynomial using Ferrari's method, and then works backward from four possible solutions to $cos(\theta)$ to arrive at the four ambiguous pose solutions which solve the P3P problem.

B. Comparison method

We performed computational experiments to characterize the numerical stability of the three P3P solution methods. Computation time was also measured; however, because the comparison was between entirely different bodies of code, the computation time measurement in this section should be taken as less rigorous (as compared to results presented later, in which computation time was measured for a single body of code with and without certain modifications). Nevertheless, broad trends were observed from the computation time results.

For test problems, we generated a set of 1E5 orthogonal 4-point sets at random location within bounds, and random attitude. The location bounds applied to the location of P_1, and were: in x, -5 to 55; in y, -25 to 25; and in z, -25 to 25. A constant target geometry was used, in which the first side length (P_1,P_2) was of magnitude 1 (thus, distances were in units of this target edge length). The second edge (P_2,P_3) was of magnitude u, which could be set, determining target aspect ratio. We will show results for $u=1$ (square target) and $u=3$.

The first three points, in order, were supplied to the P3P solver. Ordinarily, the fourth point would be used for disambiguation, which is separate from the P3P solution. However, since we were only interested in the performance of the P3P solution, we "disambiguated" by the following process: find, for each of the four P3P solutions, the largest distance error between any of the four points as estimated, and their truth locations; the solution whose worst error is the smallest is the best solution, and that worst error is taken as the associated error for that trial.

As a comment, our simulation has the effect of moving a "target spacecraft" of set geometry around to different points in front of the camera. This differs from P3P computational experiment techniques used commonly, as in [21] and [5], that randomly generate x-, y- and z-coordinates of points P_1, P_2, P_3, P_4. This latter method will create targets of varying size that will tend to loom larger in the field of view of the camera (sensor, etc), which reduces the likelihood of numerical problems. This is illustrated in Fig. 15. For this reason, the mean errors for the Linnainmaa solver shown here are greater than those shown for Linnainmaa in [21], and the median errors for the Kneip solver here are greater than those shown in [5].

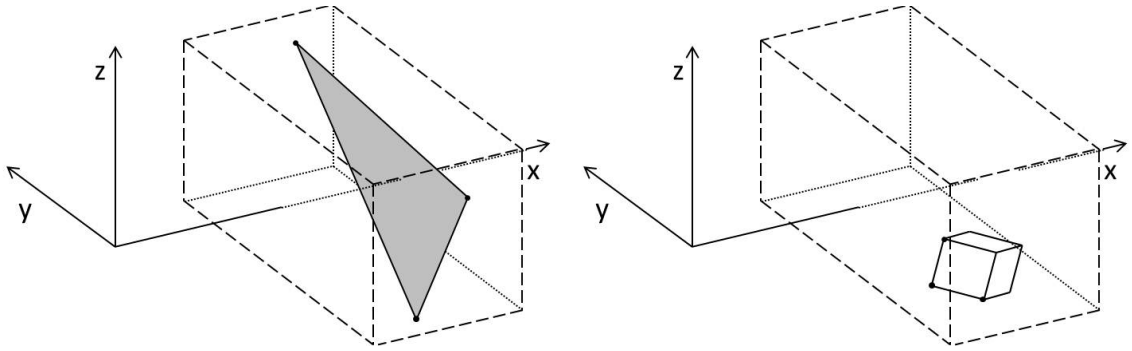

Fig. 15 Illustration of targets created by randomly selecting three points in a target space (left) vs. selecting a spacecraft location in a target space (right).

Over the 1E5 runs, median, mean and maximum error were found for each test set. CPU time to run the entire 1E5 runs was also taken; however, this CPU time involved computational overhead that may not have been optimized, and so we only took the CPU time results as general indicators, for these tests.

These computational experiments were coded and executed in MATLAB.

C. Results and discussion

Results are shown in Table 2 (square targets, aspect ratio $u=1$) and Table 3 (rectangular targets, aspect ratio $u=3$). In both cases, the data appear to be shown twice: for 1E5 trials, and then a number one or two less than that. This is because in both cases, the Linnainmaa solver failed in the number of trials indicated, and median and mean error could not be determined. To deal with this, the failed cases were removed in post-processing from each of the solvers, and results without those cases were compiled.

Table 2 Median, mean and max error for three P3P solvers, for trials with 1E3, 1E4 and 1E5 targets at randomly defined relative pose, and target aspect ratio u=1 (square).

Method	# trials	Worst distance error at a cube corner (cube side length = 1)			CPU time, s
		Median	Mean	Max	
Linnainmaa	100,000	NaN	NaN	1.69e+3	230.86
Hafer		9.15e-12	0.0039	62.60	553.64
Kneip		4.49e-12	4.95e-7	0.0110	151.81
Linnainmaa	99,998 (-2 L failure points)	4.04e-11	0.0223	1.69e+3	
Hafer		9.15e-12	0.0036	62.60	
Kneip		4.49e-12	4.95e-7	0.0110	

Table 3 Median, mean and max error for three P3P solvers, for trials with 1E3, 1E4 and 1E5 targets at randomly defined relative pose, and target aspect ratio u=3 (ratio of rectangle sides).

Method	# trials	Worst distance error at a cube corner (cube side length = 3)			CPU time, s
		Median	Mean	Max	
Linnainmaa	100,000	NaN	NaN	1.07e+3	229.23
Hafer		3.65e-12	0.0193	1.40e+3	561.06
Kneip		3.35e-12	8.74e-5	2.6366	151.30
Linnainmaa	99,999 (-1 L failure point)	1.03e-11	0.0182	1.07e+3	
Hafer		3.65e-12	0.0190	1.40e+3	
Kneip		3.35e-12	8.74e-5	2.6366	

The Kneip solver out-performs the others in both numerical stability and speed. Of particular note, while both the Linnainmaa and Hafer (original) solutions encountered cases where the error was orders of magnitude greater than the truth values, the Kneip solver did not. The Kneip solver possesses excellent numerical stability.

These findings, along with those of [5] (published in 2011, in which Kneip compared his solver to that of [20]), indicate that the Kneip solver can be taken as a state-of-the-art P3P solver.

Given that a generic P3P solver had "beaten" the solver we made specifically for orthogonal targets, we then asked whether the generic solver could be modified to create an even better P3P solver specifically for orthogonal targets. It occurs to us in hindsight that the Kneip solver may have been well suited to adaptation for orthogonal targets. But at the time, it was for its performance benefits only that it was chosen for further study.

CHAPTER IV

TAILORED P3P SOLUTION FOR ORTHOGONAL TARGETS

In this chapter, we discuss how the Kneip solver is tailored for an orthogonal target. The discussion moves from the mathematical simplifications used, to their effects on the code written to implement the algorithm, and finally to experiments assessing computation time.

A. Algorithm modification to avoid re-ordering of points

As a preliminary step, the Kneip algorithm was modified to avoid the re-ordering of points (P_1,P_2,P_3). In its present configuration, the solver conditionally re-ordered points (P_1,P_2) in order to ensure that $\theta \geq 0$. Since re-ordering (P_1,P_2) would break the orthogonality of the ordered set (P_1,P_2,P_3), the Kneip solver was modified to instead be robust to $\theta < 0$. This modification was readily made.

B. Normalization of distance

The Kneip solver was modified to find range solutions in units of distance normalized to the distance of edge (P_1,P_2), on condition that the target geometry given as input to the solver matched this convention. This step is not necessary, but reduces calculation time slightly. This reduction in computation time is accounted for separately from that due to knowledge of the orthogonal target.

The resultant position vector must be converted back into units of distance used by the spacecraft, adding three multiplication steps to the solution. This step is included in the CPU time measurements of all solutions using distance normalization.

To solve in terms of a normalized distance, two changes were made. First, the calculation of the first unit vector used in creating the N matrix from equations (3) was changed, such that the vector pointing from P_1 to P_2 was not divided by its norm, since its magnitude is now required to be 1:

$$\vec{n}_x = \frac{\left[\vec{P}_2 - \vec{P}_1\right]_w}{\left|\left[\vec{P}_2 - \vec{P}_1\right]_w\right|} \qquad \rightarrow \vec{n}_x = \left[\vec{P}_2 - \vec{P}_1\right]_w$$

$$\vec{n}_z = \frac{[\vec{n}_x \times (\vec{P}_3 - \vec{P}_1)]_w}{\left|[\vec{n}_x \times (\vec{P}_3 - \vec{P}_1)]_w\right|}$$

$$\vec{n}_y = \vec{n}_z \times \vec{n}_x$$

$$N = \left[\left[\vec{n}_x\right]_w \left[\vec{n}_y\right]_w \left[\vec{n}_z\right]_w\right]^T$$

Second, since the geometry parameter d_{12}, the distance P_1 to P_2, was required to be *1*, d_{12} was removed when used in multiplication (and its replacement with *1* in addition). This removed a parameter from the coordinates of the target in the target frame:

$$[P_1]_{\eta'} = [\ 0 \quad 0 \quad 0\ ]^T;$$

$$[P_2]_{\eta'} = [\ d_{12} \quad 0 \quad 0\ ]^T; \qquad \rightarrow [P2]_{\eta'} = [\ 1 \quad 0 \quad 0\ ]^T;$$

$$[P_3]_{\eta'} = [\ p_1 \quad p_2 \quad 0\ ]^T;$$

This also simplified the calculation of the coefficients of the 4th order polynomial, which constitutes a large portion of the solver's work. As we will see, additional simplifications for the orthogonal geometry further simplified these coefficients.

The formulae for the polynomial coefficients, for the unmodified solver, the distance-normalized solver, and two cases of the orthogonal solver, can be seen by skipping forward to Table 4 (pg. 45).

C. Tailored Kneip solution for orthogonal targets

Given the distance normalized Kneip solver, it requires few steps to build the assumption of an orthogonal geometry. The steps are:

1. Modify the transformation matrix between the world and target frames. The modifications reflect the fact that the separation distance between points P_2 and P_1 is taken to be 1 (normalized); similarly, that between P_3 and P_2 is u, the rectangle aspect ratio; and these $\vec{P}_2$ and $\vec{P}_3$ are orthogonal.

$$\vec{n}_x = \left[\vec{P}_2 - \vec{P}_1\right]_w$$

$$\vec{n}_z = \frac{[\vec{n}_x \times (\vec{P}_3 - \vec{P}_1)]_w}{\left|[\vec{n}_x \times (\vec{P}_3 - \vec{P}_1)]_w\right|} \qquad \rightarrow \vec{n}_y = \frac{\left[\vec{P}_3 - \vec{P}_2\right]_w}{u}$$

$$\vec{n}_y = \vec{n}_z \times \vec{n}_x \qquad \vec{n}_z = \vec{n}_x \times \vec{n}_y$$

$$N = \left[\left[\vec{n}_x\right]_w \left[\vec{n}_y\right]_w \left[\vec{n}_z\right]_w\right]^T$$

2. Define geometry parameters given perpendicularity:

$$[\mathrm{P}_1]_{\eta'} = [\ 0 \qquad 0 \qquad 0\ ]^{\mathrm{T}};$$

$$[\mathrm{P}_2]_{\eta'} = [\ 1 \qquad 0 \qquad 0\ ]^{\mathrm{T}};$$

$$[\mathrm{P}_3]_{\eta'} = [\ p_1 \qquad p_2 \qquad 0\ ]^{\mathrm{T}}; \quad \rightarrow [\mathrm{P3}]_{\eta'} = [\ 1\ u \qquad 0\ ]^{\mathrm{T}};$$

In other words, we set $p_1=1$ and $p_2=u$.

These differences between the N frame geometry for the case of a general target and an orthogonal target are illustrated in Fig. 16.

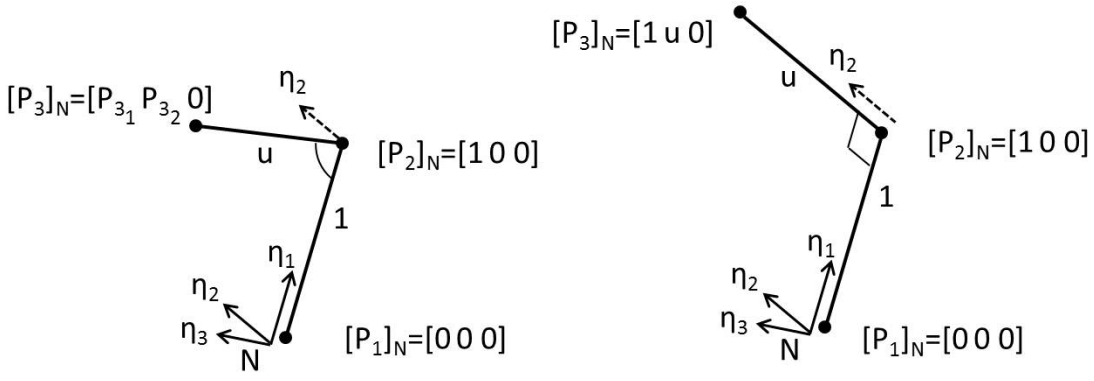

Fig. 16 Target geometry in $_N$ frame for a general target (left) and an orthogonal target (right).

These simplifications further simplify the coefficients of the 4th order polynomial created by the Kneip solver to find the pose solutions.

D. Kneip solution for square targets

We further considered the special case of the orthogonal target, when the aspect ratio is 1, and the target is square. (This may have some relevance given that many popular cubesats are square.) To further tailor the solver for a square target, we again modify the target geometry parameters, as:

$$\vec{n}_x = \left[\vec{P}_2 - \vec{P}_1\right]_w$$

$$\vec{n}_y = \frac{\left[\vec{P}_3 - \vec{P}_2\right]_w}{u} \qquad \rightarrow \vec{n}_y = \left[\vec{P}_3 - \vec{P}_2\right]_w$$

$$\vec{n}_z = \vec{n}_x \times \vec{n}_y$$

$$N = \left[\left[\vec{n}_x\right]_w \left[\vec{n}_y\right]_w \left[\vec{n}_z\right]_w\right]^T$$

and

$$[P_1]_{\eta'} = [\ 0 \quad 0 \quad 0\ ]^T;$$

$$[P_2]_{\eta'} = [\ 1 \quad 0 \quad 0\ ]^T;$$

$$[P_3]_{\eta'} = [\ 1 \quad u \quad 0\ ]^T; \quad \rightarrow [P_3]_{\eta'} = [\ 1 \quad 1 \quad 0\ ]^T;$$

In essence, all the geometry parameters are now equal to *1*, which clearly makes for a simple system.

E. Summary of computational reductions in parameters and operations

To summarize the past few sections, we present the following tables:

Table 4 shows the formulae for the polynomial coefficients a_4, a_3, a_2, a_1, a_0, in the case of the unmodified solver, and for each of the simplifying conditions discussed.

Table 5 shows the number of addition and multiplication operations required to calculate each coefficient, for each scenario. Multiplication by (-1) was not considered an operation for this purpose, because it was assumed that this was done with less computational burden than general multiplication, or addition.

Table 6 summarizes the simplifications to point coordinates, variables removed from computation, and operations removed from computation for each scenario.

Table 4 Polynomial coefficients a_4, a_3, a_2, a_1, a_0 for P3P solution, unmodified and for various simplification scenarios.

Solver, unmodified	$a_4 = -\varphi_2^2 p_2^4 - \varphi_1^2 p_2^4 - p_2^4$ $a_3 = 2 p_2^3 d_{12} b + 2 \varphi_2^2 p_2^3 d_{12} b - 2 \varphi_1 \varphi_2 p_2^3 d_{12}$ $a_2 = -\varphi_2^2 p_1^2 p_2^2 - \varphi_2^2 p_2^2 d_{12}^2 b^2 - \varphi_2^2 p_2^2 d_{12}^2 + \varphi_2^2 p_2^4 + \varphi_1^2 p_2^4 + 2 p_1 p_2^2 d_{12} + 2 \varphi_1 \varphi_2 p_1 p_2^2 d_{12} b - \varphi_1^2 p_1^2 p_2^2 + 2 \varphi_2^2 p_1 p_2^2 d_{12} - p_2^2 d_{12}^2 b^2 - 2 p_1^2 p_2^2$ $a_1 = 2 p_1^2 p_2 d_{12} b + 2 \varphi_1 \varphi_2 p_2^3 d_{12} - 2 \varphi_2^2 p_2^3 d_{12} b - 2 p_1 p_2 d_{12}^2 b$ $a_0 = -2 \varphi_1 \varphi_2 p_1 p_2^2 d_{12} b + \varphi_2^2 p_2^2 d_{12}^2 + 2 p_1^3 d_{12} - p_1^2 d_{12}^2 + \varphi_2^2 p_1^2 p_2^2 - p_1^4 - 2 \varphi_2^2 p_1 p_2^2 d_{12} + \varphi_1^2 p_1^2 p_2^2 + \varphi_2^2 p_2^2 d_{12}^2 b^2$
Normalized distance	$a_4 = -\varphi_2^2 p_2^4 - \varphi_1^2 p_2^4 - p_2^4$ $a_3 = 2 p_2^3 b + 2 \varphi_2^2 p_2^3 b - 2 \varphi_1 \varphi_2 p_2^3$ $a_2 = -\varphi_2^2 p_1^2 p_2^2 - \varphi_2^2 p_2^2 b^2 - \varphi_2^2 p_2^2 + \varphi_2^2 p_2^4 + \varphi_1^2 p_2^4 + 2 p_1 p_2^2 + 2 \varphi_1 \varphi_2 p_1 p_2^2 b - \varphi_1^2 p_1^2 p_2^2 + 2 \varphi_2^2 p_1 p_2^2 - p_2^2 b^2 - 2 p_1^2 p_2^2$ $a_1 = 2 p_1^2 p_2 b + 2 \varphi_1 \varphi_2 p_2^3 - 2 \varphi_2^2 p_2^3 b - 2 p_1 p_2 b$ $a_0 = -2 \varphi_1 \varphi_2 p_1 p_2^2 b + \varphi_2^2 p_2^2 + 2 p_1^3 - p_1^2 + \varphi_2^2 p_1^2 p_2^2 - p_1^4 - 2 \varphi_2^2 p_1 p_2^2 + \varphi_1^2 p_1^2 p_2^2 + \varphi_2^2 p_2^2 b^2$
Tailored for orthogonal target	$a_4 = -\varphi_2^2 p_2^4 - \varphi_1^2 p_2^4 - p_2^4$ $a_3 = 2 p_2^3 b + 2 \varphi_2^2 p_2^3 b - 2 \varphi_1 \varphi_2 p_2^3$ $a_2 = -\varphi_2^2 p_2^2 b^2 + \varphi_1^2 p_2^4 + \varphi_2^2 p_2^4 + 2 \varphi_1 \varphi_2 p_2^2 b - \varphi_1^2 p_2^2 - p_2^2 b^2$ $a_1 = 2 \varphi_1 \varphi_2 p_2^3 - 2 \varphi_2^2 p_2^3 b$ $a_0 = -2 \varphi_1 \varphi_2 p_2^2 b + \varphi_1^2 p_2^2 + \varphi_2^2 p_2^2 b^2$
Tailored for square target	$a_4 = -\varphi_2^2 - \varphi_1^2 - 1$ $a_3 = 2 b + 2 \varphi_2^2 b - 2 \varphi_1 \varphi_2$ $a_2 = -\varphi_2^2 b^2 + \varphi_2^2 + 2 \varphi_1 \varphi_2 b - b^2$ $a_1 = 2 \varphi_1 \varphi_2 - 2 \varphi_2^2 b$ $a_0 = -2 \varphi_1 \varphi_2 b + \varphi_1^2 + \varphi_2^2 b^2$
$\phi_1 = \frac{[f_3]_\tau^T \cdot [100]}{[f_3]_\tau^T \cdot [001]}, \phi_2 = \frac{[f_3]_\tau^T \cdot [010]}{[f_3]_\tau^T \cdot [001]}$	

Table 5 Multiplication and addition operations for calculation of P3P solver 4th order polynomial coefficients, not counting multiplication by (-1).

	a_4	a_3	a_2	a_1	a_0	Total
Solver, unmodified	4	13	38	19	30	104
Normalized distance	4	10	32	15	24	85
Tailored for orthogonal target	4	10	15	7	9	45
Tailored for square target	2	7	7	5	6	27

Table 6 Summary of mathematical simplifications, reduction in code operations, and reduction in computation time of solver, for various code configurations.

	N matrix	Parameters removed	# multiplications, additions removed
Solver, unmodified	-	-	- (total is 104)
Normalized distance	$\vec{n}_x$ not divided by its norm	d_{12}, d_{12}^2	19-3=16*
Tailored for orthogonal target	$\vec{n}_y$ calculated without cross product	p_1, p_1^2, p_1^3, p_1^4	40
Tailored for square target	$\vec{n}_y$ not divided by u	p_2, p_2^2, p_2^3, p_2^4	18
* *Three more multiplications were added to convert the components of the solved position vector into operational distance units.*			

F. Computational performance of tailored solver

In addition to the analysis of computational savings above, we performed tests which measured the computation time of the algorithms. Due to time restrictions, the tests were coded in MATLAB, which introduces more unseen computational overhead than a

lower-level language such as C. For this reason, we present these as “soft” numbers, acknowledging that implementation in C would be a useful next step.

The time measurement used was the ‘cputime’ function in MATLAB. The smallest time increment measurable by this function was 0.0156 s. For this reason, instead of measuring the time for the solver to run once, a loop was set up in which the solver was run 1E4 times, between the start and stop of the timer. This put the times measured at ~10 seconds, or 641x measurement precision.

In the first set of tests, a single P3P problem was defined, and time was measured while the problem was solved 1E4 times. Four tests were performed in these conditions, with each of the tests comparing two solvers:

1) Unmodified solver vs. distance normalized solver, solving orthogonal non-square targets
2) Distance normalized solver vs. orthogonal optimized solver, solving orthogonal non-square targets
3) Unmodified solver vs. distance normalized solver, solving square targets
4) Distance normalized solver vs. square optimized solver, solving square targets

To avoid experimental contamination due to other processor tasks, an additional step was taken: ten iterations of each comparison of the two solvers were performed. So, the test followed the form of:

- Solve the P3P problem with the first solver, 1E4 times, and measure the time.
- Solve again with the second solver, 1E4 times, and measure the time.
- Return to the first solver and repeat. Repeat each test ten times, alternating.

- Average the results of the ten runs for each solver.

This process of alternating between solvers was used to average out any effects of the general state of the processor (i.e. its load due to other computer functions). Table 7 shows the results these tests.

A second test was performed, in which 100 different P3P problems were defined. This was done to examine whether there was anything unique about the particular P3P problem used in the previous tests. Other than that, the test was identical, and so this test took 100 times longer to run. This test was only run for comparison 2) above, the distance-normalized solver vs. the orthogonal optimized solver. The results of this test are shown in Table 8

It was observed that the first solution typically took more time than subsequent solutions. The cause for this was not clear, but was assumed to be related to MATLAB internal operations. The effect of this was expected to be insignificant due to the multiple repetitions of each test.

Table 7 Computational time measurements, with a single problem solved 1E4 times sequentially.

	Solving:	Mean time, 1 run	% reduction
		ms	
Unmodified	orthogonal targets, *u=3*	1.02	-
Normalized distance		0.93	7.2%
Normalized distance (2)		0.93	-
Tailored for orth.		0.54	42.0%
Unmodified	Square targets, *u=1*	1.03	-
Normalized distance		0.95	6.9%
Normalized distance (2)		0.94	-
Tailored for orth.		0.54	42.4%

Table 8 Computational time measurements, with 1E4 sequential computations of each problem, for 100 unique problems solved.

	Solving:	Mean time, 1 run	% reduction
		Ms	
Normalized distance	orthogonal targets, *u=3*	0.94	-
Tailored for orth.		0.54	42.4%

The results indicate a reduction in computation time of over 40% for the solver that is built specifically to solve for an orthogonal target. This is beyond the 7% reduction that occurred only by framing the problem in terms of normalized distance units, which is a separate technique but also one that is applicable to any P3P solver.

When tests are extended from a single P3P problem to 100 different problems in the case of the orthogonal solver, the time savings results are essentially unchanged.

While the analytical comparison performed in the previous section suggested that the solver for square targets offered additional computational benefit beyond the solver for generic orthogonal targets, there appears to be very little additional benefit achieved in practice. We do not know now whether this could be explained by better analysis of the computational task as it was implemented in MATLAB, or whether it would be futile to examine further without moving to C, where exactly what is being computed can be better known.

G. Discussion

We consider this finding to have two implications. First, if one wishes to design a P3P solver for a known orthogonal target, such as the described relative navigation system for a cubesat, it is likely that the modified Kneip algorithm we have constructed here will be a very good choice from among all P3P solvers. The number of P3P solvers, and the lack (to our knowledge) of a recent comparison of them, makes a confident statement about the state of the art impossible. It seems it would again be of value, as Haralick et al. did in 1991 [21], to provide a comprehensive survey of the field. What we can say is: the orthogonal Kneip P3P solver identified here is computationally faster, for problems with orthogonal targets, than the unmodified Kneip solver; and that solver is at least as fast, and much more numerically stable, than both the Linnainmaa solver, and the original solver also developed here; and also that according to Kneip's own findings, his solver is numerically faster and more stable than another contemporary benchmark, that of [20].

The second implication is that the work done here to incorporate knowledge of target geometry into the Kneip solver could be undertaken both for other P3P solvers, and other target geometries, if desired. In this way, we see these findings as a general indication that if one can be certain about the geometry of the targets a PNP-type vision system is to see, then it should be possible to improve performance by incorporating knowledge of those geometries into the solver. To our knowledge, this has not been explicitly considered before. Knowledge of target geometry is clearly not possible for many PNP problems. However, other researchers (for example, [25]) have suggested that there are navigation problems, other than those of cooperative spacecraft, when target geometry can be specified in advance.

CHAPTER V

TARGET GEOMETRY APPLIED TO POINT CLOUD CORRESPONDENCE FOR IMPROVED SOLUTION

We have so far discussed a means of optimizing the computational requirements for solving the P3P problem, given an orthogonal target geometry. As described at the outset of this thesis, the P3P problem results in up to 4 ambiguous solutions. To complete the relative pose solution, these must be disambiguated using one or more additional control features. In addition, measurement error should be accounted for. To complete the line of questioning of this thesis, we have selected a means of proceeding from the P3P solution not accounting for measurement error, to a least squares estimate, as seen previously in the literature. The method selected was a PCC technique. Then, in a similar fashion as before, we modified its geometry parameterization to reflect a known geometry, and examined the resulting computational benefit. An orthogonal geometry was used in this study; however, as will be discussed, the methods used here could also be used to tailor the PCC solver to a specific geometry even if it was not orthogonal.

We break the discussion into the following aspects. First, we describe how solving a point cloud correspondence problem can be used to estimate a solution, although it may not be optimal. We also describe how to frame the results of the P3P solution as an input to the PCC problem. We then describe a method of solving the PCC problem using the Optimal Linear Attitude Estimator (OLAE), which achieves very low computational

burden. We then show how the OLAE method can be tailored given an orthogonal target, analogous to the approach taken with the P3P solution; and we present computational performance comparisons between the tailored and general PCC (OLAE) solvers for solving problems with orthogonal targets. Finally, optimality criteria for an estimated pose solution, given measurements with error, are discussed.

A. Hand-off from P3P problem to point cloud correspondence problem

As discussed earlier, this problem seeks to find best estimated values for a rotation matrix R and a translation vector T given measurements p_i' with noise N_i of body vectors p_i, produced by:

$$p_i' = Rp_i + T + N_i$$

The reader may also refer back to Fig. 10 for an illustration of this problem.

We arrive at this problem from the solution to the P3P problem by the following route. As discussed earlier, the P3P solution will typically include 2-4 ambiguous solutions. If measurements are perfect, then measurements from the fourth point will precisely match those predicted by one of the solutions, which will be the correct solution. However, if measurements included error, measurements from the fourth point will in general not match those predicted from any of the solutions. Given this, a logical next step is to pick the solution from among the four that has the lowest measurement error corresponding to the fourth point, as a starting point for further improvement.

At this point, we have a translation vector, T, and a rotation matrix, R, relating the camera frame to the target frame, where the correspondences of the first three points

relate perfectly to their measured values under this relationship, but there is error in the fourth correspondence. This situation is illustrated in Fig. 17. A better estimate would distribute error over each point, allowing smaller error levels at any one point; i.e., a least squares approach.

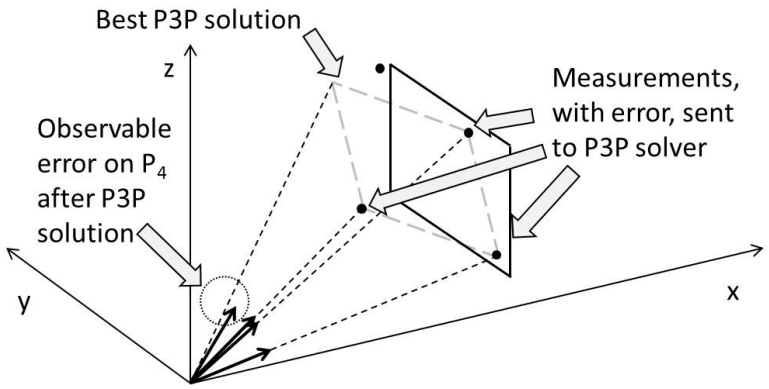

Fig. 17 State estimate after P3P solution. Observable error remains in the fourth correspondence.

Some improvement can be made by applying the PCC technique. This technique requires, as inputs, measurements of vectors p_i', which are related to target geometry vectors p_i by a translation and a rotation that will be estimated. The measured vectors p_i' are created as follows. The first three vectors for points (P_1, P_2, P_3) are taken directly from the P3P solution. However, if the vector to P_4 were taken from the P3P solution, it would a) not indicate any measurement error, and b) not correspond to actual measurements from P_4. Therefore, the vector to P_4 is taken to be along the line of the

measurement $\hat{b}_4$, with length such that the norm distance to the P3P-predicted location of P_4 is minimized. This is illustrated in Fig. 18. This vector is found by the following relationship:

$$p_4' = r\hat{b}_4$$
$$r = \vec{P}_4 \cdot \hat{b}_4$$

where $\vec{P}_4$ is the location of P_4 predicted by P3P.

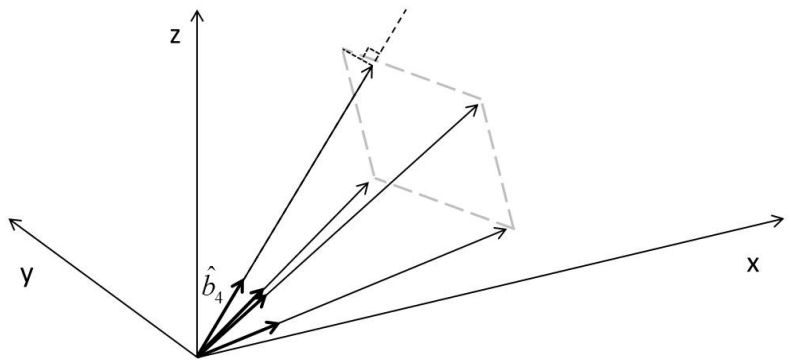

Fig. 18 Fourth vector for PCC is taken as vector along measurement $\hat{b}_4$ minimizing norm to location of P_4 predicted by P3P.

We provide no citations for this method of creating p_4', because we did not see it explained in the literature; we assume it is the method used by other researchers, such as Kneip.

With this, we have four "measured" vectors, p_i', *i=1..4*, that can be provided to the PCC solver along with known geometry vectors p_i, to arrive at improved estimates of *R* and *T*, the rotation and translation of the camera with respect to the target.

We discuss the point cloud solver used in the next section. We conclude this section by presenting, in Fig. 19, a flow chart summarizing the steps used to proceed from the P3P problem to the PCC problem, so as to obtain an improved estimate of camera location and orientation.

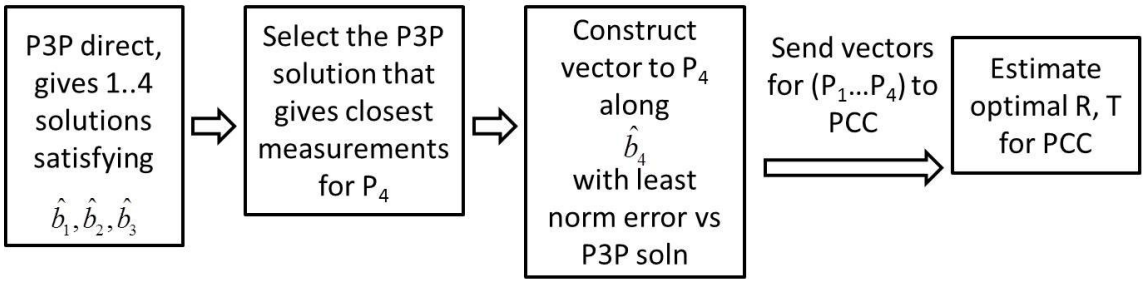

Fig. 19 Flowchart for applying PCC technique to estimate an improvement to a P3P solution.

B. PCC as an attitude estimation problem

The PCC problem is framed as an attitude-only problem by taking both measured and known geometry vectors with respect to their centroids (this discussion follows [6]):

$$q_i' = p_i' - p'$$
$$p' = \frac{1}{N}\sum_{i=1}^{N} p_i'$$

and

$$q_i = p_i - p$$
$$p = \frac{1}{N}\sum_{i=1}^{N} p_i$$

Thus, we have (momentarily ignoring noise):

$$q_i' = p_i' - p' = Rp_i + T - \frac{1}{N}\sum_{i=1}^{N} p_i' = Rp_i + T - \frac{1}{N}\sum_{i=1}^{N}\left(Rp_i + T\right)$$

$$= Rp_i + T - T - \frac{1}{N}\sum_{i=1}^{N} Rp_i = R\left(p_i - p\right)$$

$$q_i' = Rq_i \tag{5}$$

With noise, the problem is to find a least-squares estimate $\hat{R}$ that best satisfies equation (5). In our approach, this is done using OLAE (next section). Once this is done, $\hat{T}$ is found according to:

$$\hat{T} = p' - \hat{R}p$$

There are various ways of solving the PCC problem. Kneip used Arun's method, which is a direct solution using singular value decomposition (SVD) of a matrix constructed from the measured and known vectors q_i' and q_i. We employ a different method, known as the Optimal Linear Attitude Estimator (OLAE).

C. Optimal Linear Attitude Estimator

The details of the OLAE method can be found in [7] and [8]. Here, we provide an overview of the steps to achieve a solution, sufficient to understand the geometry parameterization, and tailoring for an orthogonal target geometry.

The method seeks to estimate the attitude of a target given vector position measurements (as distinct from angle, line-of-sight, etc.), still containing error, of features on the target's body. A hypothetical instance of the problem is illustrated in Fig. 20. The difference between measurement types makes this problem distinct from the PNP problem. However, as was shown above, it is equivalent to solving the attitude portion (the most difficult part) of the PCC problem.

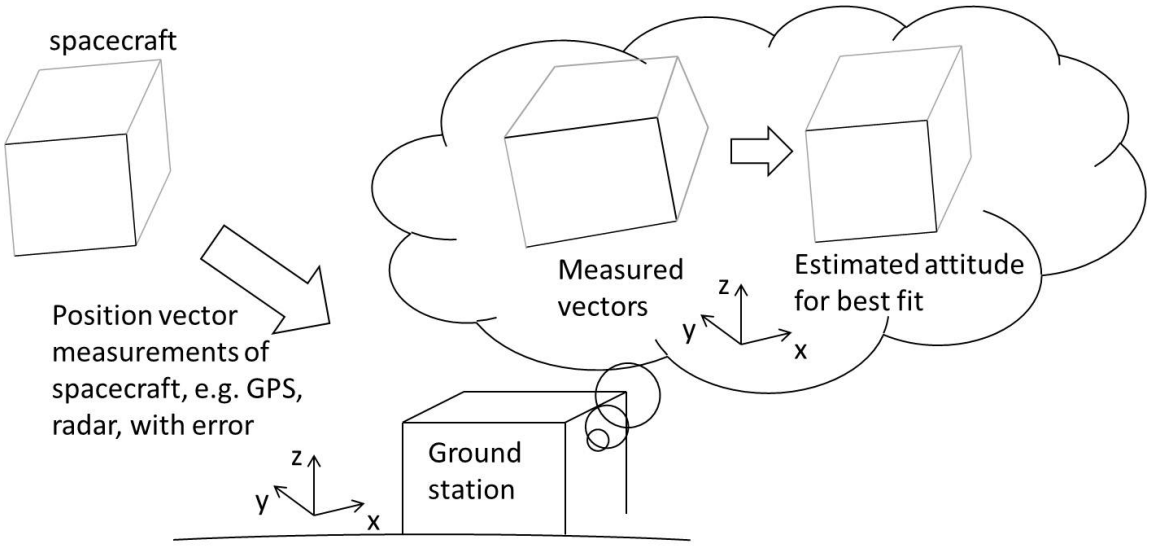

Fig. 20 Illustration of attitude estimation problem.

OLAE uses the parameterization of the rotation matrix *R* in the Classical Rodriguez Parameters (CRP's) [$g_1\ g_2\ g_3$], according to the Cayley transformation:

$$R = \left[1+G\right]^{-1}\left[1-G\right]$$

$$G = g^{\times} = \begin{bmatrix} 0 & -g_3 & g_2 \\ g_3 & 0 & -g_1 \\ -g_2 & g_1 & 0 \end{bmatrix}$$

Proceeding from equation (5), we use the Cayley Transformation, and linear algebra manipulation, to arrive at:

$$\left(q_i' - q_i\right) = -G\left(q_i' + q_i\right)$$

With shorthand for the sum and difference terms, we have:

$$q_{i_{diff}} = -Gq_{i_{sum}}$$

or

$$q_{i_{diff}} = q_{i_{sum}}{}^{\times} g \tag{6}$$

with $q_{i_{sum}}{}^{\times}$ the skew-symmetric matrix created from the vector $q_{i_{sum}}$.

If we now re-introduce noise, equation (6) becomes:

$$q_{i_{diff}} = q_{i_{sum}}{}^{\times} g + e_i$$

which is a linear relationship of the form $y = Hx + e$ between state quantities, **g**, and known or measured quantities, $q_{i_{diff}}$ and $q_{i_{sum}}$, with **e** representing measurement error. This well-known relationship can be solved to obtain a least-squares estimate of state quantities using the normal equation, $x = \left(H^T H\right)^{-1} H^T y$. Here, *x* would be the

parameters **g** of the rotation matrix $\hat{R}$. Given $\hat{R}$, we take $\hat{T}$ such that the centroids of the measured and estimated vector sets are the same: $\hat{T} = p' - Rp$.

The $\hat{R}$ and $\hat{T}$ achieved using the OLAE method, given the inputs from the selected P3P solution described earlier, are taken as final estimates of camera position and orientation.

Note: OLAE finds an optimal attitude matrix $\hat{R}$ according to an optimality criterion that is not quite the standard Wahba optimality criterion, $\sum_{i=1}^{n} \left| \hat{R}p_i - \left(p_i' - \hat{T} \right) \right|^2$. This difference (OLAE vs. Wahba optimality) is addressed in the referenced paper, and is not considered here, because the difference between using these two criteria is not expected to be significant. However, the effect of using a criterion of this general family (i.e. the Wahba or OLAE criteria) is significant, and is discussed later in this chapter.

D. Computation of OLAE solution

Geometry parameters used in the OLAE computation are as follows. Inputs are four vectors of known target geometry, and four measured vectors. Because we are considering a P4P problem in which the four target points are coplanar, the third entry in each of the known geometry vectors is always zero. The known and measured vectors are:

$$[q_i]_N = [\, q_{i_1} \;\; q_{i_2} \; 0 \,]^T$$

in $_N$, the target frame; and

$$[\, q_i' \,]_c = [\, q_{i_1}' \; q_{i_2}' \; q_{i_3}' \,]^T$$

in $_c$, the camera frame.

We then compute and store the sums and differences needed to form the vector $q_{i_{diff}}$ and the matrix $q_{i_{sum}}$:

$$q_{ij_{sum}} = q_{i_j} + q_{i_j}'$$

$$q_{ij_{diff}} = q_{i_j} - q_{i_j}'$$

for *i=1..4* and *j=1..2*, comprising sixteen sum/difference parameters.

With this done, the remaining steps can be followed:

1) Form matrix $q_{i_{sum}}{}^{\times}$ and vector $q_{i_{diff}}$
2) Compute $g_{opt} = \left[q_{sum}{}^{\times T} q_{sum}{}^{\times} \right]^{-1} q_{sum}{}^{\times T} q_{diff}$
3) Compute $\hat{R} = \left[1 + G_{opt} \right]^{-1} \left[1 - G_{opt} \right]$. G_{opt} is skew symmetric matrix of g_{opt}.
4) Compute $\hat{T} = p' - Rp$.

To hasten step 2 above, the matrix $\left[q_{sum}{}^{\times T} q_{sum}{}^{\times} \right]^{-1}$ and the vector $q_{sum}{}^{\times T} q_{diff}$ are both created directly from inputs. The matrix inversion is not performed as such. Instead, the known formula for the inverse of a 3x3 matrix, in terms of its nine entries ($m_{11}..m_{33}$), is used. The entries ($m_{11}..m_{33}$) are themselves computed as functions of the components of $q_{i_{diff}}$ and $q_{i_{sum}}$.

E. Orthogonal target geometry parameterization in OLAE

In comparison to the Kneip P3P solver, less is done to tailor the OLAE solver to an orthogonal geometry. The representations of the measured vectors remain the same:

$$[\,q_i'\,]_c = [\,q_{i_1}'\;q_{i_2}'\;q_{i_3}'\,]^T$$

and the representations of the known geometry vectors are changed from:

$$[q_I]_N = [\,q_{i_1}\;\;q_{i_2}\;0]^T$$

to (written explicitly):

$$[q_1]_N = [-0.5 \qquad -0.5\,u \qquad 0]^T$$

$$[q_2]_N = [\;0.5 \qquad -0.5\,u \qquad 0]^T$$

$$[q_3]_N = [\;0.5 \qquad 0.5\,u \qquad 0]^T$$

$$[q_4]_N = [-0.5 \qquad 0.5\,u \qquad 0]^T$$

The sum/difference parameters then become:

$$q_{ij_{sum}} = (sign)\,0.5\begin{pmatrix}1\\u\end{pmatrix} + q_{i_j}'$$

$$q_{ij_{diff}} = (sign)\,0.5\begin{pmatrix}1\\u\end{pmatrix} - q_{i_j}'$$

where the parentheses are not meant to denote a formal matrix notation, but simply to convey that sixteen equations over ($i=1..4,\ j=1..2$) are being summarized into two.

Computing the entries of $q_{i_{diff}}$ and $q_{i_{sum}}$ in the above manner is the only step taken to tailor the OLAE solver for an orthogonal geometry. The computational reductions are:

1) Avoid reading variables for the target geometry into function's memory, since they are always the same when target geometry is known;
2) Skip the step of calculating the centroid of the known geometry vectors; and,
3) Replace addition and subtraction of two variable quantities, to addition and subtraction of one variable quantity with one a priori known quantity.

We believe that item 2 could be exploited beyond what has been done in this study, by writing a bit-level algorithm to perform fast addition given a known bit sequence as one addend. We will discuss this further after we present results.

The simplifications shown here are implemented for an orthogonal target geometry, but they could just as well be implemented for another geometry. All that is required is a specific geometry that is known to the solver. This contrasts with the Kneip P3P solver, where significant further benefit was derived from the geometry being orthogonal.

F. Comparison of orthogonal and general solvers

Computational time of the PCC solver when solving problems with orthogonal geometries was measured, for unmodified solvers and solvers tailored for the orthogonal geometry. Methods similar to those used in the P3P performance study were used. MATLAB was again the coding environment. In this case, each solver was run 1E5 times iteratively between the stop and start of the cputime function (vs. 1E4 times for the P3P solvers), in order to get magnitude of time measured on the order of 10 s. As this indicates, the PCC solver performed approximately one order of magnitude faster than the P3P solver. As in the P3P case, the test module alternated ten times between the unmodified and tailored algorithms, solving the problem 1E5 times each time.

Table 9 shows results in which a single PCC problem was solved. Comparisons between a tailored and unmodified solver are made twice, once for orthogonal targets with aspect ratio=3 and a second time for square targets (aspect ratio=1).

Table 10 shows similar results in which the same test was performed 100 times, for 100 unique PCC problems, for aspect ratio=3 only. This was done to verify that results were not particular to a single PCC problem.

Table 9 OLAE computational time measurements, with a single problem solved 1E5 times sequentially.

	Solving:	Mean time, 1 run	% reduction
		ms	
Unmodified	orthogonal targets, *u=3*	0.126	-
Tailored for orth.		0.115	9.1%
Unmodified	Square targets, *u=1*	0.125	-
Tailored for orth.		0.112	10.9%

Table 10 OLAE computational time measurements, with 1E5 sequential computations of each problem, for 100 unique problems solved.

	Solving:	Mean time, 1 run	% reduction
		ms	
Unmodified	orthogonal targets, *u=3*	0.125	-
Tailored for orth.		0.114	8.8%

G. Discussion

Relative time savings due to tailoring the solver are less significant for the PCC solver than was the case for the P3P solver. We offer the following comments.

As mentioned, we did not exploit one opportunity to legitimately hasten the PCC calculation given the orthogonal geometry, by developing a fast bit-wise addition algorithm for adding one unknown quantity to a second known quantity. We did not do this primarily due to our own time restriction. However, we would not expect it to change the relative time reduction all that much, since a great deal of computational time was expended in the subsequent linear algebra steps that, it seemed, could not be optimized.

The simplifications to the PCC solver could be applied, with slight modifications, to any known geometry, while the simplifications to the P3P solver required the orthogonal geometry. This may explain why greater performance reduction was achieved with the P3P solver, which seemed well matched to the orthogonal geometry.

Overall, the computational time of the PCC solver was an order of magnitude less than that of the P3P solver. This fact might favor its use onboard an operational flight computer, despite the fact that its result is not optimal for the PNP problem. (This is discussed in the next chapter.)

Although the time consumption of the PCC solver is small compared to P3P, we still find the results of optimizing it for orthogonal targets interesting, because it augments the idea that this approach may show good results for other PNP solvers and techniques. This may provide a useful operational benefit in environments where computational resources are limited, for problems where target geometry can be known.

H. A note on optimality criteria

For clarity, we review two facts about our implementation of the OLAE/PCC solver to obtain an improved P4P solution.

First, the optimality criterion used by the OLAE solver is not the Wahba optimality criterion. This is discussed in the referenced paper, which concludes that the difference in the optimized result is not expected to be significant.

Second, any optimality criterion employed by the PCC technique (whether it be the Wahba criterion, the OLAE criterion, or other) will not be an optimal solution for the PNP problem. This is because the measurement modes are different, and because translation and rotation are coupled in the PNP problem, while in the PCC problem they are decoupled. This means that by using the Wahba optimality criterion for P3P solution improvement, the best estimate solution vectors will be constrained to have the same centroid as the “measured” vectors resulting from the P3P solution. This is shown in Fig. 21. However, these are not “measured” vectors, as the point cloud problem considers them to be; but rather, solution vectors resulting from the entirely different measurements of the P3P problem. This will not be the optimal solution in the sense of most closely matching measurements.

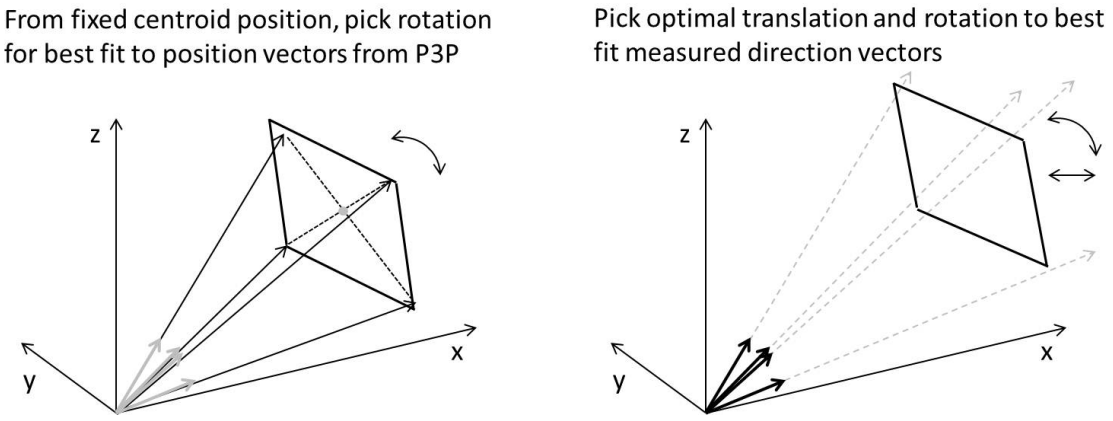

Fig. 21 Illustration of the estimate found using PCC (left), vs. an estimate minimizing error to P3P measurements (right).

An optimality criterion matched to the PNP problem, as given in [24] (where the ½ multiplier is omitted), is:

$$J = \sum_{i=1}^{n} \left| \left[\hat{b}_i\right]_c - \left[\hat{r}_i\right]_c \right|^2$$

$$\left[r_i\right]_c = \left[C\right]\left[r_i\right]_N + \left[T\right]_c$$

$$\left[\hat{r}_i\right]_c = \frac{\left[r_i\right]_c}{\left|\left[r_i\right]_c\right|}$$

$$\left[\hat{b}_i\right]_c = \left[\hat{r}_i\right]_c + \left[\eta_i\right]$$

Here, a "truth" unit vector pointing to each feature, $\hat{r}_i$, is derived, and compared to the measured unit vector $\hat{b}_i$ used in the PNP problem.

Just as a least-squares solution to the Wahba criterion can be found using direct methods, the above criterion can also be computed directly; and this is done in [24]. This amounts to a direct PNP solution that is optimal given measurements with error. This

solution would be expected to offer better accuracy than the method used here of performing a P3P solution and a PCC improvement. However, we expect that the computational time is greater as well, although we would require further study to be sure.

It would be of interest to us to examine whether the orthogonal target geometry technique could be applied to the direct PNP method cited here. This is mentioned under future work for this project.

Finally, while optimality above is considered in terms of an estimated solution that most closely matches measurements, it can also be useful to look at how closely a solution matches truth data, if such truth data is available (i.e. in an experimental setting). We have performed initial comparisons of the accuracy of the pose solution achieved using the PCC technique described here, vs. an iterative technique which, assuming convergence, ought to arrive at the optimal solution for least-squares measurement error. When compared against truth data (translation vector and rotation matrix), our expectation that the iterative solution would achieve lower error than the PCC solution has not been supported, in initial investigations. This point requires more study, both in terms of analysis, and review of any literature that may address this question. The question is important to anyone interested in applying the solutions presented here in an engineering setting, because the P3P/PCC solution technique is expected to take less computation time than the PCC/GLSDC iterative technique, and perhaps also less than other techniques, such as Hesch's direct PNP technique.

CHAPTER VI

CONCLUSIONS AND FUTURE WORK

A P3P direct solution for orthogonal target geometries is shown here, presented as an adaptation of the solution of L. Kneip, that achieves improved computational speed. Our literature search has not turned up a solver that would be expected to perform better. However, a great many P3P solvers abound, and it is a study in itself – perhaps a useful one – to review them quantitatively.

Our results with the Kneip P3P and OLAE PCC solvers suggest that some optimization for known geometries may be possible for other solvers. While this approach may only offer limited further benefit beyond the state of the art, it seems still to have some value, as otherwise performance is left on the table.

We close by identifying opportunities for future work.

As mentioned earlier, it might be of interest to apply the tailored orthogonal geometry approach to the Hesch method, or another method for a direct least squares PNP solution. It seems the Kneip P3P method may have been particularly well suited to adaptation for orthogonal geometries, although we did not choose it with any insight that this might be the case. It might be interesting, in general, to see how tailoring to orthogonal geometry would fare over a broader pool of PNP solver tools.

Just as PNP approaches can be applied to the spacecraft relative navigation with vector measurements problem, we wondered whether some of the work developed on the

attitude estimation side of this problem might be under-utilized in the PNP community. Particularly, we considered whether OLAE's use of the Cayley transformation representation of the attitude matrix could also be used to reduce the order of the attitude parameters appearing in the PNP optimality cost function defined by Hesch in [24], and if this might result in computational benefits to his solver. If that solver is state of the art (it is a recent publication), further improvements to it might be of interest among the broader community of PNP researchers.

Finally, anyone wishing to apply this work in an engineering setting should consider the accuracy required in their solution vs. the computational load of the various methods available for solving the PNP problem. To this end, accuracy studies might be performed with relatively low effort to compare the solutions achieved using various methods vs. truth data, if the studies required for a given scenario have not already been addressed in the literature.

REFERENCES

[1] Fischler, M., and Bolles, R., "Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography," *Communications of the ACM*, Vol. 24, No. 6, 1981, pp. 381–395.

[2] URL: http://www.cubesat.org/index.php/about-us/mission-statement [last accessed on 15 March 2012]

[3] Photos courtesy of www.cubesat.org [last accessed on 21 February 2012]

[4] Linnainmaa, S., Harwood, D., and Davis, L., "Pose estimation of a three-dimensional object using triangle pairs," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 10, No. 5, 1988, pp. 634–647.

[5] Kneip, L., Scaramuzza, D., and Siegwart, R., "A Novel Parametrization of the Perspective-Three-Point Problem for a Direct Computation of Absolute Camera Position and Orientation," *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2011.

[6] Arun, K., Huang, T., and Blostein, S., "Least-squares fitting of two 3-d points sets," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 9, No. 5, 1987, pp. 698–700.

[7] Mortari, D., Markley, F. L., and Junkins, J. L., "Optimal Linear Attitude Estimator," *Advances in the Astronautical Sciences*, Vol. 105, Pt. I, 2000, pp. 465–478.

[8] Mortari, D., Markley, F.L., and Singla, P., "An optimal linear attitude estimator," *Journal of Guidance, Control, and Dynamics*, Vol. 30, No. 6, 2007, pp.1619–1627.

[9] Maeland, L., "Evaluation of a Coarse Sun Sensor in a Miniaturized Distributed Relative Navigation System: An Experimental and Analytical Investigation," Ph.D. dissertation, Aerospace Engineering Dept., Texas A&M Univ., College Station, TX, 2011.

[10] Maeland, L., Perez, J., and Reed, H., "Experimental Characterization and Model Validation of a Commercial Coarse Sun Sensor for Use in a 6-DOF Relative Navigation System," *AIAA Journal of Spacecraft and Rockets* (to be submitted).

[11] Stancliffe, D., "Analysis and Design of a Test Apparatus for Resolving Near-Field Effects Associated with Using a Coarse Sun Sensor as Part of a 6-DOF Solution," M.S. thesis, Aerospace Engineering Dept., Texas A&M Univ., College Station, TX, 2010.

[12] Maeland, L., Stancliffe, D., Perez, J., and Reed, R., "An Improved Model for Coarse Sun Sensors to Include Geometric and Optical Nonlinearities," *AIAA Journal of Spacecraft and Rockets* (to be submitted).

[13] Evolution Robotics, Inc., Northstar Detector Kit: User Guide, Part number: B-M-0059. http://URL: www.evolution.com/products/northstar [last accessed on 15 March 2012]

[14] Du, J.-Y., "Vision Based Navigation System for Autonomous Proximity Operations: An Experimental and Analytical Study," Ph.D. thesis, Aerospace Engineering Dept, Texas A&M University, December 2004.

[15] Alonso, R., Du, J., Hughes, D., Junkins, J.L., and Crassidis, J., "Relative Navigation for Formation Flight of Spacecraft", *Proceedings of the Flight Mechanics Symposium*, NASA-Goddard Space Flight Center, Greenbelt, MD, June 2001.

[16] Crassidis, J., Alonso, R., and Junkins, J., "Optimal Attitude and Position Determination from Line-of-Sight Measurements," *The Journal of the Astronautical Sciences*, Vol. 48, No. 2 and 3, 2001, pp. 391–408.

[17] Mortari, D., Rojas, M., and Junkins, J., "Attitude and Position Estimation from Vector Observations," *AAS/AIAA Space Flight Mechanics Meeting*, Paper AAS 04-140, Maui, Hawaii, February 8–12, 2004.

[18] Mukundan, R., and Ramakrishnan, K., "A Quaternion Based Solution to the Pose Determination Problem for Rendezvous and Docking Simulations," *Mathematics and Computers in Simulation*, Vol. 39, 1995, pp. 143-153.

[19] Woffinden, D. C. and Geller, D. K., "Relative Angles-Only Navigation and Pose Estimation For Autonomous Orbital Rendezvous," *Journal of Guidance, Control and Dynamics*, Vol. 30, No. 5, 2007, pp. 1–23.

[20] Gao, X., Hou, X., Tang, J., and Cheng, H., "Complete solution classification for the perspective-three-point problem," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 25, No. 8, 2003, pp. 930–943.

[21] Haralick, R., Lee, C., Ottenberg, K., and Nolle, M., "Analysis and solutions of the three point perspective pose estimation problem," *IEEE International Conference on Computer Vision and Pattern Recognition*, Maui, USA, 1991.

[22] Haralick, R., "Determining camera parameters from the perspective projection of a rectangle," *Pattern Recognition*, Vol. 22, No. 3, 1989, pp. 225–230.

[23] Penna, M., "Determining Camera Parameters from the Perspective Projection of a Quadrilateral," *Pattern Recognition*, Vol. 24, No. 6, 1991, pp. 533–541.

[24] Hesch, J. and Roumeliotis, S., "A direct least-squares (dls) solution for PnP," *Proceedings of the International Conference on Computer Vision*, Barcelona, Spain, November 6-13, 2011.

[25] Lee, R., Lu, P-C., and Tsai, W-H., "Robot Location Using Single Views of Rectangular Shapes," *PE&RS*, Vol. 56, No. 2, 1990, pp. 231-238.

[26] Tang, J., Chen, W., and Wang, J., "A Study on the P3P Problem," Lecture Notes in Computer Science, Vol. 5226/2008, 2008, pp. 422–429.

[27] Wong, K. W., "Basic mathematics of photogrammetry," *Manual of Photogrammetry*, 4th edition, edited by C. C. Slama, ASP Publishers, Falls Church, USA, 1980, pp. 37–101.

VITA

Mr. William Thomas Hafer received his undergraduate education at the Massachusetts Institute of Technology in Cambridge, MA, where he studied Mechanical Engineering. After graduation, he worked for two engineering research and development companies, Foster-Miller, Inc. (now QinetiQ North America), and Infoscitex Corp., from 2003 to 2010, in positions of Staff and Project Engineer. During this time, he served as Program Manager and Principal Investigator of government SBIR and STTR research contracts, and developed technical focus areas in spacecraft thermal control, and integration of experimental components and subsystems onto Unmanned Aerial Vehicles (UAV's). He joined the Department of Aerospace Engineering at Texas A&M University as a graduate student in the fall of 2010, where he plans to continue for his PhD studies.

Name:	William Thomas Hafer
Address:	H.R. Bright Building, Rm. 701, Ross Street - TAMU 3141 College Station, TX 77843-3141
Email Address:	willh@alum.mit.edu
Education:	B.S., Mechanical Engineering, Massachusetts Institute of Technology, 2003