Solving the Perspective-2-Point Problem for Flying-Camera Photo Composition

Ziquan Lan[1,2] David Hsu[1,2] Gim Hee Lee[2]
[1]NUS Graduate School for Integrative Sciences and Engineering,
[2]Department of Computer Science, National University of Singapore
{ziquan, dyhsu, gimhee.lee}@comp.nus.edu.sg

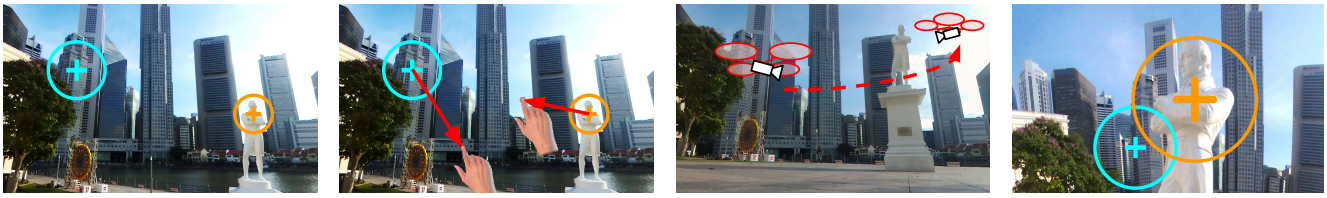

Figure 1: Flying-camera photo composition with two objects of interest: an envisioned use case. **LEFT**: The initial viewpoint. Colored circles indicate objects of interest. **MIDDLE-LEFT**: Compose a photo with simple gestures on the viewfinder image. **MIDDLE-RIGHT**: The camera determines the desired viewpoint and flies to it autonomously. **RIGHT**: The final viewpoint with the desired composition.

Abstract

Drone-mounted flying cameras will revolutionize photo-taking. The user, instead of holding a camera in hand and manually searching for a viewpoint, will interact directly with image contents in the viewfinder through simple gestures, and the flying camera will achieve the desired viewpoint through the autonomous flying capability of the drone. This work studies a common situation in photo-taking, i.e., the underlying viewpoint search problem for composing a photo with two objects of interest. We model it as a Perspective-2-Point (P2P) problem, which is under-constrained to determine the six degrees-of-freedom camera pose uniquely. By incorporating the user's composition requirements and minimizing the camera's flying distance, we form a constrained nonlinear optimization problem and solve it in closed form. *Experiments on synthetic data sets and on a flying camera system indicate promising results.*

1. Introduction

Drone-mounted flying cameras will be ubiquitous in the near future and revolutionize photo-taking. They produce fascinating photos from viewpoints unreachable previously [4]. People may bring along compact, portable flying cameras [2, 3, 6] and use their touchscreen mobile phones as viewfinders to take photos [3, 7]. The autonomous flying capability of drone-mounted cameras opens a new way of photo-taking. With traditional cameras, the user holds a camera in hand and manually searches for a favorable viewpoint. With flying cameras, the user will interact directly with the image through simple gestures on the touchscreen viewfinder, and the camera will determine the desired viewpoint and fly to it autonomously (Fig. 1). To achieve this, we study the underlying viewpoint search problem for composing a photo with two objects of interest, which is common in, *e.g.*, foreground-background or side-by-side composition.

Assume that the positions of two objects of interest are known [15, 22, 25] and their corresponding projections on the image plane are specified by the user. Determining the six degrees-of-freedom (DoFs) camera pose that satisfies the composition constraints is the well-known Perspective-n-Point (PnP) problem, with $n = 2$. P2P is related to the family of minimal problems in computer vision, and our solution approach shares a similar line of thought. While many minimal problems in computer vision attempt to solve for the unique solution of the unknown camera parameters with a minimum number of correspondences, the P2P problem is under-determined and has infinite solutions: the minimum number of point correspondences required to solve the PnP problem is 3. We propose to solve the problem by adding the constraint that the flying camera should reach one of the solutions as fast as possible along the shortest path. This leads to a constrained nonlinear optimization problem.

We show that our constrained nonlinear optimization problem can be solved in *closed form*. By analyzing the geometry of the solution space and applying the first-order optimality conditions of the objective function, we derive

a maximum of 16 candidate solutions. We then obtain the global optimum by enumeration. While generic nonlinear optimization methods can solve our problem as well, they are not practical on-board the drones that have limited computation power. In addition, they often get stuck in local minima, because of the nonlinear constraints. In summary, we introduce an interesting instance of the PnP problem with $n = 2$ for flying-camera photo composition (Section 3). We provide a closed-form solution to the resulting constrained nonlinear optimization problem (Section 4). Finally, we conduct experiments on synthetic data sets and on a flying camera system to evaluate our solution for feasibility and robustness (Section 5).

2. Related Work

2.1. User Interactions for Photo Composition

Photo composition with flying cameras is usually achieved by explicitly controlling the camera pose using, *e.g.*, joystick controllers and GPS maps [3, 8]. Joysticks allow the user to control the pan, tilt, dolly, truck and pedestal motions of the camera. However, controlling the low-level camera motions is not only tedious, but also often results in loss of situational awareness, inaccurate attitude judgment, and failure to detect obstacles [26]. Compared with joystick controllers, GPS maps enable task-level camera motions, such as tracking [3, 5], orbiting [1, 5], and navigating to waypoints [3, 7]. However, GPS maps do not supply the visual information to identify attractive viewpoints for visual composition.

An alternative is to directly interact with the image content. Through-the-lens camera control has been proposed for cameras in virtual environments [17, 21]. It allows the user to directly manipulate objects' locations on the screen, so that the camera parameters are automatically recomputed to satisfy the desired on-screen locations. Such direct manipulation techniques have been successfully demonstrated in real-time flying camera system [22], which proposed a sample-based method to compute the camera pose. However, its solutions vary from time to time depending on the random samples. In contrast, this study aims to determine a unique viewpoint by minimizing the flying distance.

2.2. Minimal Problems

Minimal problems in computer vision are the problems solved from a minimal number of correspondences. For example, in the five-point relative pose problem [29], five corresponding image points are needed to provide five epipolar equations to estimate the essential matrix. A sixth point correspondence is needed if the focal length is unknown [31] or there is a radial distortion parameter to be estimated [11, 20]. Similarly, additional point correspondences are required if there are more unknown camera parameters, such as the eight-point problem for estimating fundamental matrix and single radial distortion parameter for uncalibrated cameras [19], and the nine-point problem for estimating fundamental matrix and two different distortion parameters for uncalibrated cameras [11, 20]. In contrast, our problem solves for a camera pose with six unknown parameters, but it only has four equations derived from two correspondences.

2.3. Perpsective-n-Point

PnP problems estimate the rotation and translation of a calibrated perspective camera by using n known 3D reference points and their corresponding 2D image projections. Since each correspondence provides two equality constraints, the minimal case is having three correspondences [16]. Four and more correspondences have been also investigated to improve robustness [23, 24, 32, 34, 35].

This study solves a P2P problem. Early studies on P2P make additional assumptions, such as planar motion constraints [10, 13], known camera orientation [9, 27], known viewing direction and triangulation constraint of the 3D points [12]. We assume no such assumptions. Instead, we form an optimization problem to find the solution with the minimal flying distance to reach.

3. Problem Formulation

Composition in photography is usually referred to as the placement of relative objects in an image frame [18]. Each object is an abstract notion of how the photographer interprets a group of visual elements in the scene. We represent each object as a ball B_j^W, $j = 1, 2$, in a world coordinate frame F_W. The ball center $\mathbf{q}_j^W = [x_j^W \; y_j^W \; z_j^W]^T$ represents the estimated object centroid position, and the radius ϵ_j denotes a collision-free distance estimated based on the object size during selection [22]. Correspondingly, the user-specified object projections are represented as $\mathbf{p}_j^I = [u_j^I \; v_j^I \; 1]^T$ in the homogeneous image coordinate frame F_I. In addition, the camera has a starting camera position $\mathbf{t}_0^W = [x_0^W \; y_0^W \; z_0^W]^T$ in F_W. We are interested in a camera pose in F_W that could be represented as a rotation matrix $\mathbf{R}_C^W$ and a translation vector $\mathbf{t}_C^W$ from F_C to F_W. The resulting camera pose should be the nearest one that not only satisfies desired object compositions but also not colliding with the two objects. Hence, we formulate an optimization problem as follows.

$$\underset{\mathbf{R}_C^W, \mathbf{t}_C^W}{\operatorname{argmin}} \|\mathbf{t}_C^W - \mathbf{t}_0^W\|^2, \tag{1}$$

subject to

$$\lambda_j \mathbf{p}_j^I = \mathbf{K}\,(\mathbf{R}_W^C\,\mathbf{q}_j^W + \mathbf{t}_W^C), \tag{2a}$$

$$\|\mathbf{t}_C^W - \mathbf{q}_j^W\| \geq \epsilon_j, \tag{2b}$$

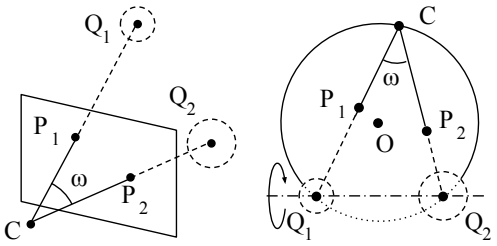

Figure 2: Camera positions that satisfy constraints in Eq. (2). C denotes the camera's center of perspective. Q_1 and Q_2 denote the object positions. P_1 and P_2 denote the user-specified object projections. **LEFT**: $\angle Q_1CQ_2 = \angle P_1CP_2 = \omega$ is a constant angle. **RIGHT**: All points on the solid arc of $\odot O$ satisfy constraints in Eq. (2). Rotating the arc around the axis through Q_1 and Q_2 forms a toroid, on which all points satisfy constraints in Eq. (2) as well.

in which $j = 1,2$. λ_j denotes the depth factor of the j-th object. $\mathbf{K}$ is the calibrated intrinsic parameter matrix for the perspective camera with the pinhole imaging model. $\mathbf{R}_W^C$ and $\mathbf{t}_W^C$ denote the rotation matrix and translation vector from F_W to F_C, respectively.

The equality constraint in Eq. (2a) corresponds to the projective imaging equations derived from the two point correspondences. Since $\mathbf{K}$ is known, it is convenient to use the normalized camera coordinate frame

$$\hat{\mathbf{p}}_j^C = [\hat{u}_j^C \ \hat{v}_j^C \ 1]^T = \mathbf{K}^{-1}\,\mathbf{p}_j^I,\ j = 1,2. \tag{3}$$

The inequality constraint in Eq. (2b) ensures a collision-free distance to each object. It is reasonable to assume that $\epsilon_1 + \epsilon_2$ is relatively small as compared to the distance between the two objects, which, in fact, ensures the existence of the solution.

4. P2P Solution in Closed Form

Before diving into the details of the algebraic derivation, we first analyze the solution space of the constraints. The solution space for camera positions $\in \mathscr{R}^3$ that satisfy the constraints is a toroidal surface (see Fig. 2 and 3). Each camera position corresponds to a unique camera orientation $\in SO(3)$, so that we can first solve for the position and then determine the orientation.

Parameterizing the toroidal solution space $\in \mathscr{R}^3$ is the key to solve the problem. We introduce an auxiliary frame with one axis passing through the rotational axis of the toroid for easy parameterization.

The remaining part of the section solves the optimization problem with the following steps. First, we construct the auxiliary coordinate frame. Then, we reformulate the constraints and the objective function using the auxiliary frame. Finally, we solve the equation system derived from the first-order optimal condition of the objective function to find the global optimal camera position, hence, the corresponding camera orientation.

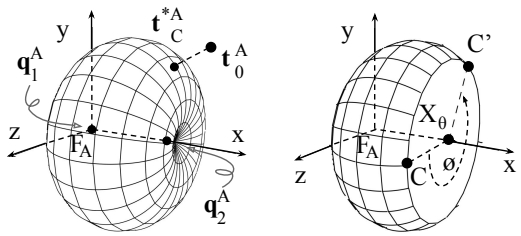

Figure 3: The auxiliary frame F_A. **LEFT**: The objective is to find $\mathbf{t}^{*A}_C$ on the toroid with the minimal distance to the initial camera position $\mathbf{t}_0^A$. **RIGHT**: The intersections between the toroid and any plane $x = X_\theta$ in F_A are circles, so any point C' on the circle can be parametrized using a reference point C and an angle ϕ.

4.1. Constructing the Auxiliary Frame

We construct the auxiliary frame F_A with one axis be the axis of rotation passing through $\mathbf{q}_1^W$ and $\mathbf{q}_2^W$ as shown in Fig. 2. More specifically, $\mathbf{q}_1^W$ coincides with the origin of F_A and $\mathbf{q}_2^W$ sits on the positive x-axis of F_A, see Fig. 3,

$$\mathbf{q}_1^A = \begin{bmatrix}0\\0\\0\end{bmatrix},\ \mathbf{q}_2^A = \begin{bmatrix}\xi_{len}\\0\\0\end{bmatrix},\ \xi_{len} = \|\mathbf{q}_1^W - \mathbf{q}_2^W\|. \tag{4}$$

Using the auxiliary frame, we manage to reformulate and solve the original problem in a simpler form, as shown in the subsequent subsections.

Let $\mathbf{R}_A^W$ and $\mathbf{t}_A^W$ be the rotation matrix and translation vector from F_A to F_W, respectively. Then, $\mathbf{t}_A^W = \mathbf{q}_1^W$, and the first column of $\mathbf{R}_A^W$, $\mathbf{c_1}_A^W = (\mathbf{q}_2^W - \mathbf{q}_1^W)/\xi_{len}$. The second and the third columns of $\mathbf{R}_A^W$ could be chosen as an arbitrary pair of orthonormal vectors that span the null space of $\mathbf{c_1}_A^W$. In the case when $\mathbf{R}_A^W$ is an improper rotation matrix, *i.e.*, $\det(\mathbf{R}_A^W) = -1$, we flip the second and the third columns of $\mathbf{R}_A^W$.

4.2. Reformulating Equality Constraints

Now, we use the auxiliary frame F_A to reformulate the equality constraints in Eq. (2a) as

$$\lambda_j \hat{\mathbf{p}}_j^C = \mathbf{R}_A^C\,\mathbf{q}_j^A + \mathbf{t}_A^C,\ j = 1,2, \tag{5}$$

in which $\mathbf{R}_A^C$ and $\mathbf{t}_A^C$ are the unknown rotation matrix and translation vector from F_A to F_C, respectively. It is desirable to eliminate the depth factors as follows,

$$[\hat{\mathbf{p}}_j^C]_\times\,(\mathbf{R}_A^C\,\mathbf{q}_j^A + \mathbf{t}_A^C) = \mathbf{0}_{3\times1},\ j = 1,2, \tag{6}$$

where $[\hat{\mathbf{p}}_j^C]_\times$ is the skew-symmetric matrix for vector $\hat{\mathbf{p}}_j^C$. Eq. (6) produces six equality constraints, and two of them are redundant. We transform the remaining four equality constraints into a matrix form,

$$\mathbf{A}\,\mathbf{w} = \mathbf{0}_{4\times1}, \tag{7}$$

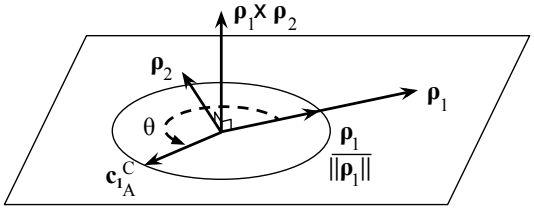

Figure 4: Parameterization of $\mathbf{c_1}_A^C$ using θ.

where

$$\mathbf{A} = \begin{bmatrix} 0 & 0 & 0 & 0 & -1 & \hat{v}_1 \\ 0 & 0 & 0 & 1 & 0 & -\hat{u}_1 \\ 0 & -1 & \hat{v}_2 & 0 & -1/\xi_{len} & \hat{v}_2/\xi_{len} \\ 1 & 0 & -\hat{u}_2 & 1/\xi_{len} & 0 & -\hat{u}_2/\xi_{len} \end{bmatrix}, \quad (8a)$$

$$\mathbf{w} = \begin{bmatrix} r_{11} & r_{21} & r_{31} & t_1 & t_2 & t_3 \end{bmatrix}^T, \quad (8b)$$

in which r_{11}, r_{21}, r_{31}, t_1, t_2 and t_3 are the entries in $\mathbf{R}_A^C$ and $\mathbf{t}_A^C$, respectively, *i.e.*,

$$\mathbf{R}_A^C = \begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix}, \ \mathbf{t}_A^C = \begin{bmatrix} t_1 \\ t_2 \\ t_3 \end{bmatrix}. \quad (9)$$

Note that there is a constraint imposed on the first three entries, r_{11}, r_{21} and r_{31}, in $\mathbf{w}$, as they form the first column $\mathbf{c_1}_A^C$ of $\mathbf{R}_A^C$, which means $\|\mathbf{c_1}_A^C\| = 1$.

The benefit of constructing F_A shows up. Since $\mathbf{q}_1^A$ and $\mathbf{q}_2^A$ are constructed to have only one non-zero entry, $\mathbf{w}$ is much simplified so that it does not contain the entries in the second column $\mathbf{c_2}_A^C$ and the third column $\mathbf{c_3}_A^C$ of $\mathbf{R}_A^C$. Otherwise, without F_A, $\mathbf{w}$ would contain all the 9 entries in a rotation matrix with more constraints imposed.

Next, we parameterize $\mathbf{w}$. The vector $\mathbf{w}$ of unknowns belongs to the null space of $\mathbf{A}$. Since $nullity(\mathbf{A}) = 2$ as $rank(\mathbf{A}) = 4$ according to Eq. (8a), $\mathbf{w}$ can be expressed as the following linear combination form,

$$\begin{aligned} \mathbf{w} &= \begin{bmatrix} \mathbf{e}_1 & \mathbf{e}_2 \end{bmatrix} \begin{bmatrix} \alpha_1 \\ \alpha_2 \end{bmatrix}, \\ \mathbf{e}_1 &= \begin{bmatrix} \hat{u}_2 & \hat{v}_2 & 1 & 0 & 0 & 0 \end{bmatrix}^T, \\ \mathbf{e}_2 &= \begin{bmatrix} \hat{u}_2 - \hat{u}_1 & \hat{v}_2 - \hat{v}_1 & 0 & \hat{u}_1\xi_{len} & \hat{v}_1\xi_{len} & \xi_{len} \end{bmatrix}^T, \end{aligned} \quad (10)$$

where α_1 and α_2 are two coefficients, $\mathbf{e}_1$ and $\mathbf{e}_2$ are the two eigenvectors of $\mathbf{A}$ corresponding to the two null eigenvalues of $\mathbf{A}$, which could be easily verified. Hence, $\mathbf{c_1}_A^C$ and $\mathbf{t}_A^C$ can be expressed as linear combinations as well,

$$\mathbf{c_1}_A^C = \begin{bmatrix} \boldsymbol{\rho}_1 & \boldsymbol{\rho}_2 \end{bmatrix} \begin{bmatrix} \alpha_1 \\ \alpha_2 \end{bmatrix}, \ \mathbf{t}_A^C = \begin{bmatrix} \boldsymbol{\tau}_1 & \boldsymbol{\tau}_2 \end{bmatrix} \begin{bmatrix} \alpha_1 \\ \alpha_2 \end{bmatrix}, \quad (11)$$

in which $\boldsymbol{\rho}_1 = \hat{\mathbf{p}}_2^C$, $\boldsymbol{\rho}_2 = \hat{\mathbf{p}}_2^C - \hat{\mathbf{p}}_1^C$, $\boldsymbol{\tau}_1 = \mathbf{0}_{3\times 1}$ and $\boldsymbol{\tau}_2 = \hat{\mathbf{p}}_1^C \xi_{len}$.

Now, $\mathbf{c_1}_A^C$ and $\mathbf{t}_A^C$ are parameterized with two parameters, α_1 and α_2. Next, we use the constraint, $\|\mathbf{c_1}_A^C\| = 1$, to reduce to a form using only one parameter. The idea is depicted in Fig. 4. We observe that $\mathbf{c_1}_A^C$ is a unit vector in the 2D plane spanned by $\boldsymbol{\rho}_1$ and $\boldsymbol{\rho}_2$. Hence, it can be parameterized using a rotation angle θ. We denote the matrix for a rotation by an angle of θ around $\boldsymbol{\rho}_1 \times \boldsymbol{\rho}_2$ as $\mathbf{R}_\theta$. Hence, $\mathbf{c_1}_A^C = \mathbf{R}_\theta \boldsymbol{\rho}_1 / \|\boldsymbol{\rho}_1\|$, which contains θ as the only parameter. More explicitly,

$$\mathbf{c_1}_A^C = \sin(\theta) \begin{bmatrix} \xi_1 & \xi_2 & \xi_3 \end{bmatrix}^T + \cos(\theta) \begin{bmatrix} \xi_4 & \xi_5 & \xi_6 \end{bmatrix}^T, \quad (12)$$

in which ξ_k's are constant, $k = 1,2,3,4,5,6$. We omit their full expressions here for brevity.

Since $\mathbf{c_1}_A^C$ is a linear combination of $\boldsymbol{\rho_1}$ and $\boldsymbol{\rho_2}$ (see Eq. (11)), we have $[\alpha_1 \ \ \alpha_2]^T = \mathbf{S}^+ \ \mathbf{c_1}_A^C$, in which $\mathbf{S}^+$ is the Moore-Penrose matrix inverse of $[\boldsymbol{\rho}_1 \ \ \boldsymbol{\rho}_2]$. Plugging $[\alpha_1 \ \ \alpha_2]^T$ back into Eq. (11), we parameterize $\mathbf{t}_A^C$ with θ as

$$\mathbf{t}_A^C = \sin(\theta) \frac{\|\boldsymbol{\rho}_1\|}{\|\boldsymbol{\rho}_1 \times \boldsymbol{\rho}_2\|} \boldsymbol{\tau_2}. \quad (13)$$

4.3. Reformulating Inequality Constraints

Similarly, we also reformulate the inequality constraints in Eq. (2b) using F_A as

$$\|\mathbf{t}_C^A - \mathbf{q}_j^A\| \geq \epsilon_j, \ j = 1,2, \quad (14)$$

where $\mathbf{t}_C^A$ is the unknown translation vector from F_C to F_A,

$$\mathbf{t}_C^A = \begin{bmatrix} x \\ y \\ z \end{bmatrix} = -\begin{bmatrix} (\mathbf{c_1}_A^C)^T \\ (\mathbf{c_2}_A^C)^T \\ (\mathbf{c_3}_A^C)^T \end{bmatrix} \mathbf{t}_A^C, \quad (15)$$

in which $\mathbf{c_1}_A^C$, $\mathbf{c_2}_A^C$ and $\mathbf{c_3}_A^C$ are respectively the first, second and third columns of $\mathbf{R}_A^C$ defined previously.

To obtain the boundary points that satisfy both the equality and inequality constraints, we use the parameterizations in the previous subsection to parameterize x, y and z in $\mathbf{t}_C^A$ respectively, as follows.

First, we can parameterize x with θ using the parameterizations of $\mathbf{c_1}_A^C$ and $\mathbf{t}_A^C$ in Eq. (12) and Eq. (13),

$$x = \xi_{len} \sin(\theta) [\sin(\theta) - \frac{\hat{\mathbf{p}}_1^C \cdot \hat{\mathbf{p}}_2^C}{\|\hat{\mathbf{p}}_1^C \times \hat{\mathbf{p}}_2^C\|} \cos(\theta)], \quad (16)$$

in which $\cdot$ and $\times$ denote the inner product and the cross product of two vectors receptively. Let ω denote the angle between the two known vectors $\hat{\mathbf{p}}_1^C$ and $\hat{\mathbf{p}}_2^C$, which corresponds to $\angle P_1 C P_2$ as shown in Fig. 2. Since $\hat{\mathbf{p}}_1^C \cdot \hat{\mathbf{p}}_2^C = \|\hat{\mathbf{p}}_1^C\| \|\hat{\mathbf{p}}_2^C\| \cos(\omega)$ and $\|\hat{\mathbf{p}}_1^C \times \hat{\mathbf{p}}_2^C\| = \|\hat{\mathbf{p}}_1^C\| \|\hat{\mathbf{p}}_2^C\| \sin(\omega)$, we can simplify x as

$$\begin{aligned} x &= \xi_{len} \sin(\theta) [\sin(\theta) - \cot(\omega) \cos(\theta)], \\ &= -\xi_{len} \csc(\omega) \sin(\theta) \cos(\theta + \omega). \end{aligned} \quad (17)$$

Next, we parameterize y and z. Since $\|\mathbf{t}_C^A\| = \|\mathbf{t}_A^C\|$, we have

$$\begin{aligned} y^2 + z^2 &= \|\mathbf{t}_A^C\|^2 - x^2, \\ &= \xi_{len}^2 \csc^2(\omega)\sin^2(\theta)\sin^2(\omega+\theta). \end{aligned} \tag{18}$$

The benefit of constructing F_A shows up, again. Eq. (17) and Eq. (18) show that both x and y^2+z^2 are functions of θ. In other words, a fixed θ determines a plane in F_A with a fixed x value, and the points on that plane forms a circle with a fixed radius $\sqrt{y^2+z^2}$, which is depicted in Fig. 3. Hence, we introduce a new parameter ϕ for the angle of rotation around the x-axis of F_A, so that

$$\begin{aligned} y &= \sin(\phi)\xi_{len}\sqrt{\csc^2(\omega)\sin^2(\theta)\sin^2(\omega+\theta)}, \\ z &= \cos(\phi)\xi_{len}\sqrt{\csc^2(\omega)\sin^2(\theta)\sin^2(\omega+\theta)}. \end{aligned} \tag{19}$$

Finally, plugging Eq. (17) and Eq. (19) back into Eq. (14), we obtain two equations for the boundary

$$\sin(\theta) = \pm\sin(\omega)\epsilon_1/\xi_{len}, \tag{20a}$$

$$\sin(\theta+\omega) = \pm\sin(\omega)\epsilon_2/\xi_{len}, \tag{20b}$$

from which we can solve for eight possible pairs of $\sin(\theta)$ and $\cos(\theta)$ corresponding to eight solutions of θ.

In fact, the parameterization of y^2+z^2 in Eq. (18) suffices our need to parameterize the boundary points. Nevertheless, the individual parameterization of y and z is useful in the next subsection.

4.4. Reformulating the Objective Function

At last, we reformulate the objective function in Eq. (1) using F_A as well,

$$\underset{\mathbf{R}_C^W, \mathbf{t}_C^W}{\operatorname{argmin}} \|\mathbf{t}_C^A - \mathbf{t}_0^A\|^2 \tag{21}$$

in which $\mathbf{t}_0^A = \mathbf{R}_A^{W\,T}(\mathbf{t}_0^W - \mathbf{t}_A^W) = [\xi_x \;\; \xi_y \;\; \xi_z]^T$ is known. This reformulated objective function essentially means minimizing the distance from the initial camera position $\mathbf{t}_0^A$ to the toroid $\mathbf{t}_C^A$ in F_A, as shown in Fig. 3.

Plugging Eq. (17) and Eq. (19) into Eq. (21), we successfully parameterize the objective function

$$\begin{aligned} obj(\phi,\theta) =& (-\xi_{len}\csc(\omega)\sin(\theta)\cos(\theta+\omega) - \xi_x)^2 \\ &+ (\sin(\phi)\sqrt{\csc^2(\omega)\sin^2(\theta)\sin^2(\omega+\theta)} - \xi_y)^2 \\ &+ (\cos(\phi)\sqrt{\csc^2(\omega)\sin^2(\theta)\sin^2(\omega+\theta)} - \xi_z)^2. \end{aligned} \tag{22}$$

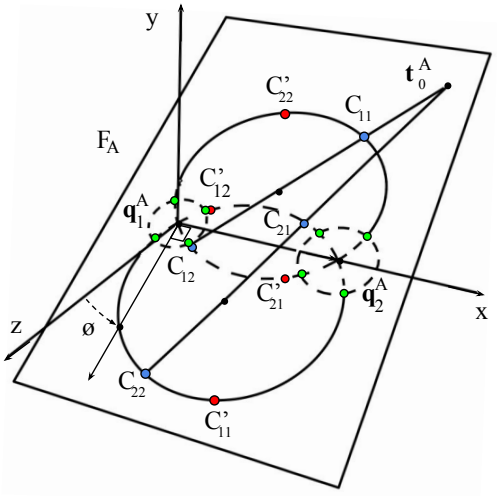

Figure 5: Optimal camera position candidates. $\mathbf{q}_1^A$, $\mathbf{q}_2^A$ and $\mathbf{t}_0^A$ determine a plane in F_A, which intersects with the toroid and forms two solid arcs. C_{11}, C_{12}, C_{21} and C_{22} in blue are four candidates. Their symmetric points about the x-axis, C'_{11}, C'_{12}, C'_{21} and C'_{22} in red, are also candidates. The green points around $\mathbf{q}_1^A$ and $\mathbf{q}_2^A$ are eight candidates at the boundary.

4.5. Minimization

To minimize *obj* with respect to ϕ and θ, we build the first-order optimality condition of Eq. (22), and identify all the stationary points and boundary points. First, we calculate the derivative of *obj* with respect to ϕ

$$\begin{aligned} \frac{\partial obj}{\partial \phi} = & 2\xi_{len}(\xi_z\sin(\phi) - \xi_y\cos(\phi)) \\ & \sqrt{\csc^2(\omega)\sin^2(\theta)\sin^2(\theta+\omega)}. \end{aligned} \tag{23}$$

Setting it to be 0, we have the following three cases

$$\sin(\theta) = 0, \tag{24a}$$

$$\sin(\theta+\omega) = 0, \tag{24b}$$

$$\sin(\phi) = \pm\frac{\xi_y}{\sqrt{\xi_y^2+\xi_z^2}}, \quad \cos(\phi) = \pm\frac{\xi_z}{\sqrt{\xi_y^2+\xi_z^2}}. \tag{24c}$$

Note that Eq. (24a) and Eq. (24b) correspond to the cases when $\mathbf{q}_1^A$ and $\mathbf{q}_2^A$ are optimal solution candidates respectively, which clearly violate the inequality constraints in Eq. (14). Suppose the problem does not have the inequality constraints, $\mathbf{q}_1^A$ and $\mathbf{q}_2^A$ will be physically infeasible solutions. Then, other optimal solution candidates solved from Eq. (24c) may not contain the true optimum.

Eq. (24c) corresponds to the general cases. It shows that ϕ only depends on the initial camera position $\mathbf{t}_0^A$. In fact, the angle ϕ corresponds to the plane determined by the three points, $\mathbf{t}_0^A$, $\mathbf{q}_1^A$ and $\mathbf{q}_2^A$, as shown in Fig. 5. Although two solutions of ϕ can be solved from Eq. (24c),

they differ from each other by π, which essentially correspond to the same plane. Therefore, the optimal camera position sits on the plane determined by $\mathbf{t}_0^A$, $\mathbf{q}_1^A$ and $\mathbf{q}_2^A$.

Next, we substitute the two solutions of ϕ from Eq. (24c) into the derivative of *obj* with respect to θ, and obtain the following two cases.

$$\frac{\partial obj}{\partial \theta} = \frac{\xi_{len}\csc^2(\omega)}{2E_1}(E_1E_2 + E_3E_4), \tag{25a}$$

$$\frac{\partial obj}{\partial \theta} = \frac{\xi_{len}\csc^2(\omega)}{2E_1}(E_1E_2 - E_3E_4), \tag{25b}$$

where E_k's are expressions, $k = 1,2,3,4$. We omit the full expressions of E_2,E_3,E_4 for brevity. We show E_1 here in order to further simplify the expressions,

$$\begin{aligned} E_1 &= \sqrt{\csc^2(\omega)\sin^2(\theta)\sin^2(\theta+\omega)}, \\ &= \pm\csc(\omega)\sin(\theta)\sin(\theta+\omega), \end{aligned} \tag{26}$$

which removes the square root by considering two cases.

Setting Eq. (25a) and Eq. (25b) to be 0, we obtain four equations, among which two are duplicates, therefore, redundant. The remaining two cases can be further simplified to

$$\sin(2\theta)\xi_7 + \cos(2\theta)\xi_8 = 0, \tag{27a}$$

$$\sin(2\theta)\xi_9 + \cos(2\theta)\xi_{10} = 0, \tag{27b}$$

where ξ_k's are constant, $k = 7,8,9,10$. Again, we omit their full expressions for brevity.

Using Eq. (27), we can easily solve for eight possible pairs of $\sin(\theta)$ and $\cos(\theta)$ corresponding to eight general solutions of θ. Fig. 5 depicts the geometric meanings of the eight general solutions.

4.6. Identifying the Best Camera Position

Plugging $\sin(\phi)$, $\cos(\phi)$, $\sin(\theta)$ and $\cos(\theta)$ solved from Eq. (24c), Eq. (20) and Eq. (27) back into Eq. (17) and Eq. (19), we can get eight solutions at the boundary and eight general solutions of $\mathbf{t}_\mathbf{C}^\mathbf{A}$ as illustrated in Fig. 5. We apply three steps to identify the best camera position in F_W. First, we use the property of $\angle \mathbf{p}_1^A \mathbf{t}_C^A \mathbf{p}_2^A = \omega$ (see Fig. 2) and the inequality constraints to eliminate the infeasible solutions. Second, among the feasible solutions, we pick ${\mathbf{t}^*}_C^A$, the one with the shortest distance to $\mathbf{t}_0^A$. Finally, we compute the best camera position ${\mathbf{t}^*}_C^W$ in F_W, using $\mathbf{R}_A^W$ and $\mathbf{t}_A^W$ constructed.

4.7. Determining the Camera Orientation

The optimal orientation is uniquely determined. In Fig. 2, the lengths of the line segments CP_1 and CP_2 are both fixed, so the optimal camera position ${\mathbf{t}^*}_C^W$ determines two more points in F_W. It is then easy and standard to retrieve the optimal camera orientation ${\mathbf{R}^*}_C^W$ using these three known points [33].

4.8. Discussion

At the end of this section, we briefly discuss the issues related to multiple optimal solutions and the change in the flying distance caused by noisy inputs.

There are three special cases with multiple optimums. First, when the two object projections coincide with each other, *i.e.*, $\omega = 0$, $\csc(\omega)$ in Eq. (17) is undefined. Then, x can be any arbitrary value and $y = z = 0$. This corresponds to the cases that the optimal camera positions are aligned with the two objects. Second, when the initial camera position is aligned with the two objects, *i.e.*, $\xi_y = \xi_z = 0$, $\sin(\phi)$ and $\cos(\phi)$ in Eq. (24c) are undefined. Then, ϕ can be any arbitrary value. The optimal camera positions form a circle. Third, when the initial camera position coincides with some point like O in Fig. 2, *i.e.*, $\xi_x = \xi_{len}/2$ and $\|\mathbf{t}_A^C\| = \csc(\omega)\xi_{len}/2$. Then, solving Eq. (27) may yield infinitely many solutions of θ, which correspond to all the points on the solid arc in Fig. 2.

Using Fig. 5, we can intuitively see how the flying distance is affected after perturbing the inputs. There are three cases. First, if $\mathbf{t}_0^A$ stays outside the toroid before and after perturbing the inputs, the toroid's shape and size are not substantially changed, so that the flying distance does not change much either. Second, if $\mathbf{t}_0^A$ stays inside the toroid before and after perturbing, the flying distance is upper bounded by the toroid size, so that the flying distance is also stable. Third, if $\mathbf{t}_0^A$ crosses the toroid surface because of the perturbation, the flying distance is small hence stable. The next section presents the empirical results of using noisy inputs.

5. Experiments

Though the closed-form solution is optimal, we are interested in its applicability and robustness in real phototaking scenarios, which contain both inaccurate human inputs caused by *e.g.*, fat-finger errors [30] and imperfect robot executions.

In this section, we first show that the resulting flying distance is robust to inaccurate user inputs. We measure the relative flying distance error using synthetic data sets. Then, we measure the quality of the resulting composition using the reprojection error with a real flying robot.

5.1. Evaluation using Synthetic Data

We use synthetic data sets to show that the flying distance is robust to inaccurate user inputs.

First, we generate ground truth data. We randomly generate 10,000 different sets of inputs. Each set consists of an initial camera position $\mathbf{t}_0^W$, two object centroids $\mathbf{q}_1^W$ and $\mathbf{q}_2^W$, two collision-free distances ϵ_1 and ϵ_2, and two object projections $\mathbf{p}_1^I$ and $\mathbf{p}_2^I$ on the image plane. The image size is 1920×1080. $\mathbf{t}_0^W$, $\mathbf{q}_1^W$ and $\mathbf{q}_2^W$ are uniformly

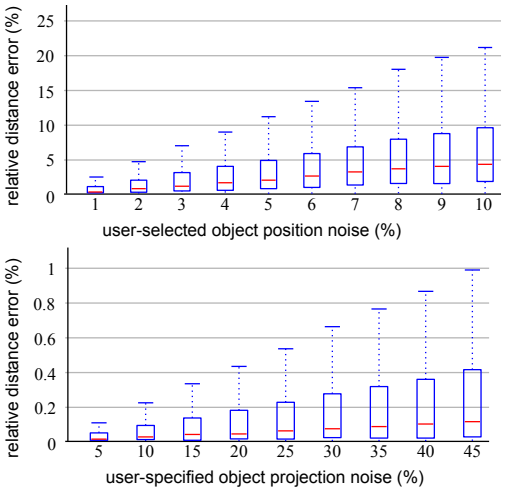

Figure 6: Results from synthetic data. The horizontal red lines indicate the medians. The boxes denote the first Q1 and third Q3 quartiles, the dashed lines extending upwards depict the statistical data extent taken to be Q3 + 1.5(Q3-Q1).

drawn from a unit cube $[0,1]^3$. ϵ_1 and ϵ_2 are uniformly generated so that $\epsilon_1 + \epsilon_2 \leq \|\mathbf{q}_1^W - \mathbf{q}_2^W\|$. $\mathbf{p}_1^I$ and $\mathbf{p}_2^I$ are uniformly drawn within the image size.

Then, we consider two types of noise introduced by inaccurate user inputs.

5.1.1 Noisy User-Selected Object Positions

During object selection, the object positions may be inaccurately estimated. We corrupt $\mathbf{q}_1^W$ and $\mathbf{q}_2^W$ with noise that follows a uniform distribution around the original object centroids. The noise ranges from 1% to 10% of their original relative distances in 3D.

5.1.2 Noisy User-Specified Object Projections

During direct manipulation, the object projections may not be specified accurately as in one's mind. We corrupt $\mathbf{p}_1^I$ and $\mathbf{p}_2^I$ with noise that follows a uniform distribution around the original object projections. The noise ranges from 5% to 45% of their original relative distances on the image to ensure non-overlapping sampling regions.

5.1.3 Measurement

We measure the relative flying distance error denoted as $Error_{dist}(\%) = |d_{noisefree} - d_{noisy}| / \|\mathbf{q}_1^W - \mathbf{q}_2^W\|$, in which $d_{noisefree} = \|\mathbf{t}_0^W - \mathbf{t}_{noisefree_C}^W\|$ is the distance from the initial camera position $\mathbf{t}_0^W$ to the optimized final position $\mathbf{t}_{noisefree_C}^W$ based on noise-free user inputs, $d_{noisy} = \|\mathbf{t}_0^W - \mathbf{t}_{noisy_C}^W\|$ is the distance from $\mathbf{t}_0^W$ to the optimized position $\mathbf{t}_{noisy_C}^W$ based on corrupted user inputs, and $\|\mathbf{q}_1^W - \mathbf{q}_2^W\|$ indicates the scale of the environment. Fig. 6 shows that

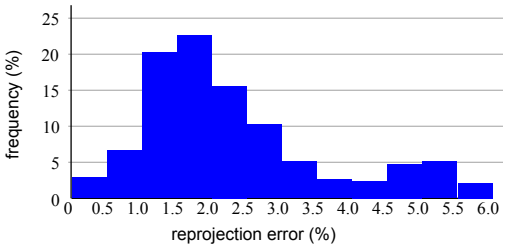

Figure 7: Results from real robot experiments.

the flying distance solved using our method is robust to very noisy user inputs in both cases.

5.2. Evaluation with Real Robot

5.2.1 Robot System Setup

The flying robot is connected via Wi-Fi to an ASUS laptop PC with an Intel Core i7 processor. We use an open-source drone driver, Bebop Autonomy [8], to communicate with the Bebop at 30 Hz. Although the camera resolution is 1920×1080, the driver [8] only transmits images of size up to 1280×720. The Bebop runs low-level controllers on-board. The PC handles all computations and hosts the Robot Operating System framework, which is used to communicate with the robot.

5.2.2 Robot Trajectory Execution

The 6 DoFs camera pose is estimated using ORB-SLAM [28] without other external sensors, such as a motion capture system. However, the drone driver [8] only supports 5 DoFs camera control. We enable the 6th DoF by rotating the image plane, so that the effective image area becomes a circular area at the center of the image, as in Fig. 8. Due to the hardware constraint, the pan-tilt range is also limited, so that some viewpoints are not achievable.

The camera flying trajectory is modeled as a straight line connecting the initial camera position to the goal position, by assuming the environment to be free of major obstruction for flying. The camera orientation is computed based on its position at that moment, in order to minimize the reprojection error at all times.

To execute the flying trajectory, each DoF of the camera is controlled by an independent PID controller. The control gains are fine-tuned by assuming that the SLAM map is in the metric unit. To resolve the scale ambiguity of the SLAM map, we adopt a scale estimation method using the on-board ultrasonic altitude measurement [14].

In practice, the goal camera viewpoint can never be achieved. A viewpoint is considered as *successful*, if it satisfies all the following conditions. The camera position error is below 0.5 meter. The orientation error is below 1 degree. All control signals at that moment are below certain thresholds as well.

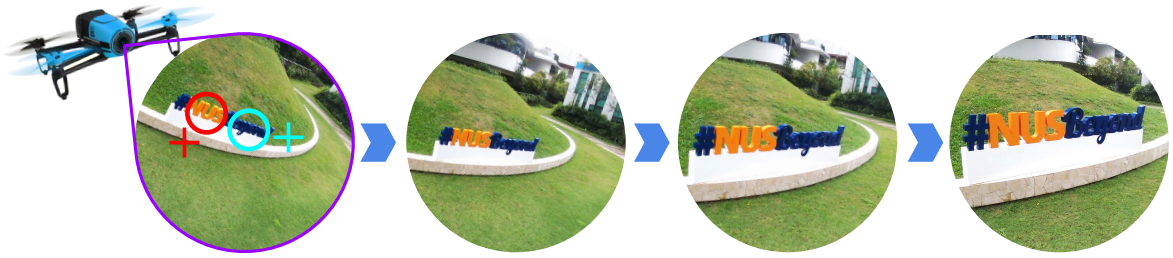

Figure 8: The viewpoint changing process. **LEFT**: In the initial viewpoint, each colored circle annotates an object that is composed to the position indicated by the same colored cross. **MIDDLE-LEFT**: The viewpoint rotates to compose one object first. **MIDDLE-RIGHT**: The viewpoint moves so that the other object gradually converges to the desired composition . **RIGHT**: The final viewpoint.

5.2.3 Data Collection

The experiments are conducted in an indoor environment of size $8m \times 6m \times 2.5m$. We use two distinctive colored balls of diameter $10cm$ to represent two objects of interest. We place the two balls at fixed locations in the center area of the room. Their distance is 1.5 meter.

We collect data from 50 trials. After building a SLAM map, we place the camera at the locations of the two balls to record their positions $\mathbf{q}_1^W$ and $\mathbf{q}_2^W$ in the SLAM map. Since the objects in real photo-taking scenarios have various shapes, we are not interested in the accuracy of object centroid estimation. We set the collision-free distances ϵ_1 and ϵ_2 to be $0.5m$ and $0.8m$ respectively.

For each trial, we randomly generate $\mathbf{t}_0^W$ within the environment. We also randomly generate $\mathbf{p}_1^I$ and $\mathbf{p}_2^I$ that are uniformly drawn from the circular area of diameter 700 pixels at the center of the image. We collect 30 frames at the successful viewpoints from each trial. We reject the unreachable viewpoints that are outside the environment or beyond the camera pan-tilt range limits.

5.2.4 Measurement

We measure the reprojection error on each collected frame. For each frame, we exact the centroid pixel $\mathbf{p}_{exacted}{}_j^I$ for each ball j, and measure the distance to its corresponding desired composition, $e_j = \|\mathbf{p}_{exacted}{}_j^I - \mathbf{p}_j^I\|$. Hence, the reprojection error of a frame with respect to the image size is $Error_{reproj}(\%) = (e_1 + e_2)/(2 \times 720)$.

The histogram in Fig. 7 shows the distribution of reprojection errors. Overall, the reprojection errors are relatively small, under very challenging conditions, including imperfect controls, SLAM system errors, etc.

5.3. Viewpoint Changing Process

We illustrate the viewpoint changing process while composing two objects using a concrete scenario. As shown in Fig. 8, the scenario starts from a high-angle viewpoint. The two main parts of the sculpture are selected as the objects of interest. We build the SLAM map, and record the estimated object positions. We manually specify the desired compositions for the objects in order to emulate the end points of direct manipulation gestures.

The viewpoint changes in two steps. First, it quickly changes the orientation to compose one object, since changing orientation is much faster than flying. Gradually, the other object converges to the desired composition while the camera is moving towards the goal. This phenomenon is caused by the fact that the orientation is recomputed based on the camera position at that time in order to minimize the reprojection error.

6. Conclusion

The evaluation results show that our closed-form solution to the formulated optimization problem is applicable to real flying robot systems. Our problem focuses on a subset of aspects that are important to flying camera photography, including minimizing the flying distance, and avoiding collisions with objects of interest. We believe the formation could be generalized to include other aspects as well, such as stabilizing the frame by removing one DoF from rotation, or collision-free flying by adding constraints for obstacles in the environments.

To conclude, this paper solved the P2P problem for flying-camera photo composition. To the best of our knowledge, this is the first study that determines the camera parameters given an under-constrained system. By incorporating the user's composition requirements and minimizing the camera's flying distance, we formed a constrained nonlinear optimization problem, solved it in closed form, and evaluated the applicability in real flying camera systems. While there are many different problem formulations using other optimization objectives and constraints, this study is the first step towards the general problems of its kind.

Acknowledgement

This work is supported in part by an NUS NGS scholarship, NUS School of Computing Strategic Initiatives, and a Singapore MOE Tier 1 grant R-252-000-637-112.

References

[1] 3DR Solo. URL https://3dr.com/solo-drone.

[2] AirSelfie. URL http://www.airselfiecamera.com.

[3] DJI Drones. URL http://www.dji.com/products/drones.

[4] Dronestagram. URL http://www.dronestagr.am/2017-international-drone-photography-contest.

[5] Hexo+. URL https://hexoplus.com.

[6] Nixie. URL http://flynixie.com.

[7] Parrot Bebop Drone. URL http://global.parrot.com/au/products/bebop-drone.

[8] AutonomyLab. Autonomylab/bebop_autonomy.

[9] M. Bansal and K. Daniilidis. Geometric urban geo-localization. In *CVPR*, 2014.

[10] O. Booij, Z. Zivkovic, et al. The planar two point algorithm. *IAS technical report, University of Amsterdam*, 2009.

[11] M. Byrod, Z. Kukelova, K. Josephson, T. Pajdla, and K. Astrom. Fast and robust numerical solutions to minimal problems for cameras with radial distortion. In *CVPR*, 2008.

[12] F. Camposeco, T. Sattler, A. Cohen, A. Geiger, and M. Pollefeys. Toroidal constraints for two-point localization under high outlier ratios. In *CVPR*, 2017.

[13] S.-I. Choi and S.-Y. Park. A new 2-point absolute pose estimation algorithm under planar motion. *Advanced Robotics*, 2015.

[14] J. Engel, J. Sturm, and D. Cremers. Scale-aware navigation of a low-cost quadrocopter with a monocular camera. *RAS*, 2014.

[15] J. Fuentes-Pacheco, J. R. Ascencio, and J. M. Rendón-Mancha. Visual simultaneous localization and mapping: a survey. *AIRE*, 2012.

[16] X.-S. Gao, X.-R. Hou, J. Tang, and H.-F. Cheng. Complete solution classification for the perspective-three-point problem. *TPAMI*, 2003.

[17] M. Gleicher and A. Witkin. Through-the-lens camera control. In *SIGGRAPH*, 1992.

[18] T. Grill and M. Scanlon. *Photographic composition*. Amphoto Books, 1990.

[19] Z. Kukelova and T. Pajdla. A minimal solution to the autocalibration of radial distortion. In *CVPR*, 2007.

[20] Z. Kukelova and T. Pajdla. Two minimal problems for cameras with radial distortion. In *ICCV*, 2007.

[21] M.-H. Kyung, M.-S. Kim, and S. J. Hong. A new approach to through-the-lens camera control. *Graphical Models and Image Processing*, 1996.

[22] Z. Lan, M. Shridhar, D. Hsu, and S. Zhao. Xpose: Reinventing user interaction with flying cameras. In *RSS*, 2017.

[23] V. Lepetit, F. Moreno-Noguer, and P. Fua. Epnp: an accurate O(n) solution to the pnp problem. *IJCV*, 2009.

[24] S. Li, C. Xu, and M. Xie. A robust O(n) solution to the perspective-n-point problem. *TPAMI*, 2012.

[25] E. Marder-Eppstein. Project tango. In *SIGGRAPH Real-Time Live!*, 2016.

[26] D. E. McGovern. Experience and results in teleoperation of land vehicles. In *Pictorial Communication in Virtual and Real Environments*, 1991.

[27] L. Merckel and T. Nishida. Evaluation of a method to solve the perspective-two-point problem using a three-axis orientation sensor. In *CIT*, 2008.

[28] R. Mur-Artal, J. M. M. Montiel, and J. D. Tardos. ORB-SLAM: a versatile and accurate monocular slam system. *T-RO*, 2015.

[29] D. Nistér. An efficient solution to the five-point relative pose problem. *TPAMI*, 2004.

[30] K. A. Siek, Y. Rogers, and K. H. Connelly. Fat finger worries: how older and younger users physically interact with PDAs. In *INTERACT*, 2005.

[31] H. Stewénius, D. Nistér, F. Kahl, and F. Schaffalitzky. A minimal solution for relative pose with unknown focal length. *Image and Vision Computing*, 2008.

[32] B. Triggs. Camera pose and calibration from 4 or 5 known 3d points. In *ICCV*, 1999.

[33] S. Umeyama. Least-squares estimation of transformation parameters between two point patterns. *TPAMI*, 1991.

[34] S. Urban, J. Leitloff, and S. Hinz. MLPnP-a real-time maximum likelihood solution to the perspective-n-point Problem. *arXiv preprint arXiv:1607.08112*, 2016.

[35] Y. Zheng, Y. Kuang, S. Sugimoto, K. Astrom, and M. Okutomi. Revisiting the pnp problem: a fast, general and optimal solution. In *ICCV*, 2013.