Computers & Graphics 52 (2015) 142–154

Contents lists available at ScienceDirect

Computers & Graphics

journal homepage: www.elsevier.com/locate/cag

Special Section

Computational rim illumination of dynamic subjects using aerial robots

Manohar Srikanth [a,*], Kavita Bala [b], Frédo Durand [a]

[a] Massachusetts Institute of Technology, United States
[b] Cornell University, United States

ARTICLE INFO

Article history:
Received 27 November 2014
Received in revised form
11 March 2015
Accepted 15 March 2015
Available online 24 March 2015

Keywords:
Computational illumination
Aerial robots

ABSTRACT

Lighting plays a major role in photography. Professional photographers use elaborate installations to light their subjects and achieve sophisticated styles. However, lighting moving subjects performing dynamic tasks presents significant challenges and requires expensive manual intervention. A skilled additional assistant might be needed to reposition lights as the subject changes pose or moves, and the extra logistics significantly raises costs and time. The associated latencies as the assistant lights the subject, and the communication required from the photographer to achieve optimum lighting could mean missing a critical shot.

We present a new approach to lighting dynamic subjects where an aerial robot equipped with a portable light source lights the subject to automatically achieve a desired lighting effect. We focus on rim lighting, a particularly challenging effect to achieve with dynamic subjects, and allow the photographer to specify a required rim width. Our algorithm processes the images from the photographer's camera and provides necessary motion commands to the aerial robot to achieve the desired rim width in the resulting photographs. With an indoor setup, we demonstrate a control approach that localizes the aerial robot with reference to the subject and tracks the subject to achieve the necessary motion. In addition to indoor experiments, we perform open-loop outdoor experiments in a realistic photo-shooting scenario to understand lighting ergonomics. Our proof-of-concept results demonstrate the utility of robots in computational lighting.

© 2015 The Authors. Published by Elsevier Ltd. This is an open access article under the CC BY-NC-ND license (http://creativecommons.org/licenses/by-nc-nd/4.0/).

1. Introduction

Lighting plays a critical role in good photography. Through the careful placement of lights, photographers can define space and enhance the mood of photographs. Perhaps the most dramatic example is the use of a rim light behind the subject, which highlights silhouettes and can be used as the main light for silhouetted styles, or as an accent to separate a subject from the background. Rim lighting effects are usually associated with posed studio photography because they require careful placement of lights relative to a static subject and often involve a whole crew where assistants move the lights until the photographer is satisfied with the look of the image. Photographers increasingly try to push the envelope and deal with dynamic scenarios by having assistants track a moving subject, but this remains costly, challenging, and might require many takes. There are two central challenges in rim lighting for photography of dynamic scenes: the ability to move the lights (usually handled by assistants), and the decision of how to move them, usually made by the photographer based on what he or she sees in the viewfinder.

In this paper we propose an automated technique for rim lighting in dynamic settings with moving subjects. We combine the use of computational photography and robotically controlled light source to facilitate the use of advanced lighting effects such as rim lighting for moving subjects. Our vision is that light sources should be actuated and be able to react to movements in the scene based on feedback from the main camera to achieve a lighting specified by the photographer. In particular, we leverage recent advances in aerial robotics and their commoditization, to enable full 3D placement of light sources around the subject. We focus on the case of rim lighting because it is particularly challenging, important for artistic control, and requires precise positioning of lights.

In our scenario, the photographer holds a main camera with the goal of capturing a picture of a potentially moving subject. The photographer controls a desired rim width as a function of the look they want to achieve. An aerial robot (also known as drones, unmanned aerial vehicles, or UAVs) is responsible for the rim light

* Corresponding author.
E-mail addresses: srimano@alum.mit.edu (M. Srikanth), kb@cs.cornell.edu (K. Bala), fredo@mit.edu (F. Durand).

http://dx.doi.org/10.1016/j.cag.2015.03.007
0097-8493/© 2015 The Authors. Published by Elsevier Ltd. This is an open access article under the CC BY-NC-ND license (http://creativecommons.org/licenses/by-nc-nd/4.0/).

and reacts to the movements of the photographer and the subject to automatically position the light to achieve the desired rim width. The photographer can also specify a wider or thinner rim width and the aerial robot responds accordingly. We introduce a simple computational measure of rim width from the image seen by the photographer's camera. While this quantitative rim width is not intended to be a direct characterization of aesthetics, it provides easy control for the photographer who can increase it or decrease it until they achieve a desired look.

To enable the automatic placement of rim lights, we introduce a new aerial robot control strategy based not only on absolute localization and sensors on the aerial robot, but also on a computational characterization of rim lighting from the main camera. At a very high level, our control strategy moves the robot away from the photographer (behind the subject) to make the rim thinner, and closer to make it wider. We focus on the placement of the robot within a given horizontal plane because it is the most critical degree of freedom for rim lighting human subjects, but a similar strategy could be used to also modulate the altitude of the aerial robot. In addition to rim width, our controller needs to achieve a number of other objectives such as respecting a given distance to the subject, keeping the robot at a given height, and making sure that the light carried by the robot is directed towards the subject. Fig. 1 top row shows our results for various rim widths.

We demonstrate our approach with a prototype that relies on a small quadrotor aerial robot that weighs less than a pound (see Fig. 1, bottom left). The aerial robot we use is a low cost (about $300), lightweight quadrotor [1,2] that can easily carry a standard flash as well as a lidar unit to help track the subject. The Lidar can easily be replaced with a cheaper alternative such as a Kinect sensor. We show that the system is able to automatically adjust to subject movement and to free photographers from the labor of lighting placement, enabling free-form photography while achieving a desired rim lighting.

This paper is an extended version of the conference paper by Srikanth et al. [3] that made the following contributions:

- We demonstrate the first approach for studio quality lighting using aerial robots.
- We introduce a control approach based on a combination of rim computation from the perspective of the camera and tracking of the subject from the aerial robot.

We make the following specific contributions in this paper:

- Using numerical simulation, we show approximate monotonic behavior of rim width with reference to the light position under certain conditions.
- We present more details on the robot control system and the image compositing.
- Unlike the setup of Srikanth et al. [3], which was an indoor lab based fully automatic system, we present an outdoor open-loop setup that essentially highlights (a) design tradeoffs for aerial robot based lighting and (b) ergonomics of robotic lighting in a realistic shooting scenario.

2. Related work

Our work is focused on closed-loop real-time computational illumination of real life objects for photographic purposes. We briefly

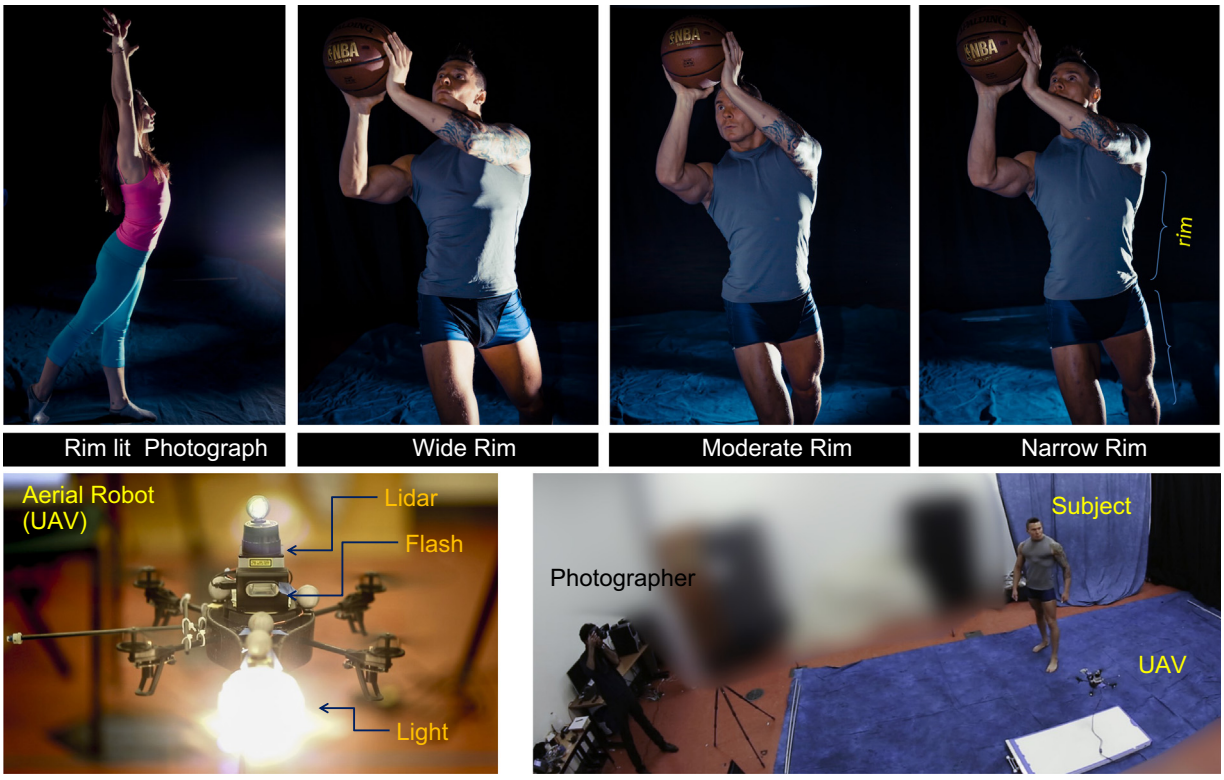

Fig. 1. By analyzing the images from the photographer's camera, an aerial robot carrying a light source automatically guides itself to a lighting location to achieve a desired rim lit photograph. *Top row*: results with various specified rim width values. *Bottom-left*: our aerial robot, a quadrotor aerial robot equipped with a flash, continuous light source, and a Lidar to detect and follow a moving subject. *Bottom-right*: our photography setup showing the subject, aerial robot and the photographer.

review papers on computational illumination of both real objects and computer graphics objects (where the geometry is known).

Computational illumination: Petschnigg et al. [4] and Eisenmann et al. [5] propose methods to computationally combine photographs, captured with and without an on-camera flash, so as to improve the overall illumination of the scene. Raskar et al. [6] propose to combine multiple photographs, each taken with a different light position, to generatea single non-photorealistic image.

Debevec et al. [7] introduced the light stage, a dome with hundreds of lights that can achieve arbitrary lighting of a dynamic subject to match a given characteristic. The main drawbacks of such an approach are cost and portability. Anrys et al. [8] propose to arrange a set of fixed lights around a near diffuse static object whose intensities are to be determined. Using an image based user interface, the user is allowed to express the desired lighting characteristics. Mohan et al. [9] propose a mechanical rotating light system that is used to capture a number of images of a static scene with various lighting conditions.With a user interface, they combine images to generate the desired lighting effect. Wang et al. [10] propose a realtime feedback system to light a dynamic scene in order to emphasize certain characteristics of the scene (for instance the edges of the objects). The main goal here is to better visualize the actual scene rather than obtaining an optimally lit photograph. Their system relies on optically aligned camera and projector through a beam splitter, and shared light spectrum (IR and visible light). More recent work by Boyadzhiev et al. [11] uses several flash photographs of a static scene captured from the same view point with varying light positions. They combine the images, with some user inputs, to produce a composite that has the desired lighting characteristics. Obviously, this approach is well suited for static scenes, for instance architectural photography. Moreover, the capture process is not well guided or automated (except for the photographer's own experience), and the photographer may end up capturing more photographs than necessary, often containing redundant information. Dorsey et al. [12,13] design stage lighting of large environments, and present a lighting control system that takes spatiotemporal issues of lighting into account. Their goal is to optimally control the installed lights in a theatrical stage.

Lighting design in computer graphics: Bousseau et al. [14] synthesize an optimal lighting environment map that enhances the appearance of scene objects from a given viewpoint. They use an image quality metric, and knowledge of the scene geometry and material properties to design the optimal environment map. Pellacini et al. [15] propose an iterative system where the user specifies the desired scene lighting by means of light painting the surface of the 3D objects. They then solve for the light parameters that achieve the desired look. Akers et al. [16] present methods to enhance rendering by changing lighting. They propose the use of rim lighting as a way to highlight the silhouette of a 3D model. Schoeneman et al. [17] present painting with light, which attempts to solve an inverse problem. Given a desired appearance of the scene that has fixed light sources, they solve for the color and the intensity of the light sources. Poulin et al. [18] propose using highlights and shadows as a user specified input; they then solve for the optimal lighting parameters.

In contrast to the above, our work is focused on real-time lighting optimization of real life dynamic objects. We use a robotic platform that responds to the motion of the subject and photographer in order to maintain a desired lighting effect. To make the robot respond quickly, we design an efficient control system for the robot. Hence, understanding and using the dynamics of the robot becomes an integral part of the problem.

Quadrotor aerial robot: A quadrotor aerial robot, also known as an unmanned aerial vehicle, is capable of complex flight motion, and is also capable of hovering at a fixed location in 3D space. Aerial robots are routinely used to carry a camera in photography and videography. To our best knowledge, application of an aerial robot to carry a light source for photographic purposes is novel. In addition, using feedback to correct and optimize the light position is also novel.

3. Overview

The photographer aims the camera at the subject, while an aerial robot equipped with a light source hovers near the subject to provide rim lighting. Our algorithm analyzes the images from the camera to evaluate the rim lighting. Given a desired rim width as specified by the photographer, our guidance policy determines the direction of motion of the robot (hence the light), so as to achieve the desired rim width. Our real-time control loop ensures quick adaptation to the changes in the subject position and posture, as well as to the changes in the position of the photographer.

The aerial robot is a standard off-the-shelf quadrotor and its four propellers provide four controllable degrees of freedom: *xyz* position, and rotation about its own vertical *z*-axis (also called yaw angle). The tilt angles (roll and pitch) are internally used for controlling the lateral *xy* position of the robot: when the robot is tilted, it moves along the corresponding horizontal direction.

We use a continuous light source and a flash strobe on the aerial robot, and both are assumed to be spot lights. The continuous light is used during the process of robot position optimization, during which it should be the dominant light source. The onboard flash strobe is triggered only at the time of capturing the final still photograph, and additional light sources can be fired as well, such as key and fill. To keep the subject in the field of the light sources, the aerial robot is always kept oriented towards the subject. We also keep it at a fixed distance to the subject to ensure safety and a constant light intensity. The location of the subject with respect to the robot is estimated using a lidar mounted on the robot.

The characteristics of the rim light as seen from the main camera are directly related to the angle around the subject between the main camera and the light source (Fig. 5). The rim light can be made thinner by moving the robot towards the back of the subject, and it can be made wider by moving it to the front. The height of the robot also affects the rim light but does not require as much adjustment in our experience, unless subjects dramatically change pose and, e.g. go from sitting to standing. In our work, we let the photographer specify the height of the robot directly and use a standard low-level control to maintain it.

Our approach is summarized in Fig. 2. We acquire video feed from the camera and compute rim width in real-time. Rim width is computed by looking at the silhouettes (gradients), that is, the width of the bright area across silhouettes (Section 4).

In each iteration, the rim width is computed and compared against the desired rim width (specified by the photographer). The difference between the measured and desired rim widths is used to determine the direction and the magnitude of the motion of the robot. The robot is moved to the new location, and the iteration is repeated until the difference is driven to zero. In addition, we track the subject location using the lidar and ensure the robot is properly oriented and positioned.

In addition to these high-level controls, we manage the subtle dynamics of the quadrotor and provide high-frequency mid and low-level control. This is a well-studied problem in robotics and we rely on mostly standard solutions: a combination of onboard sensors and absolute localization provided by a motion capture system. Future versions could replace the latter by cheaper alternatives or richer onboard sensors, but like most research on quadrotor control, we decided to focus on our high-level objectives and use the motion

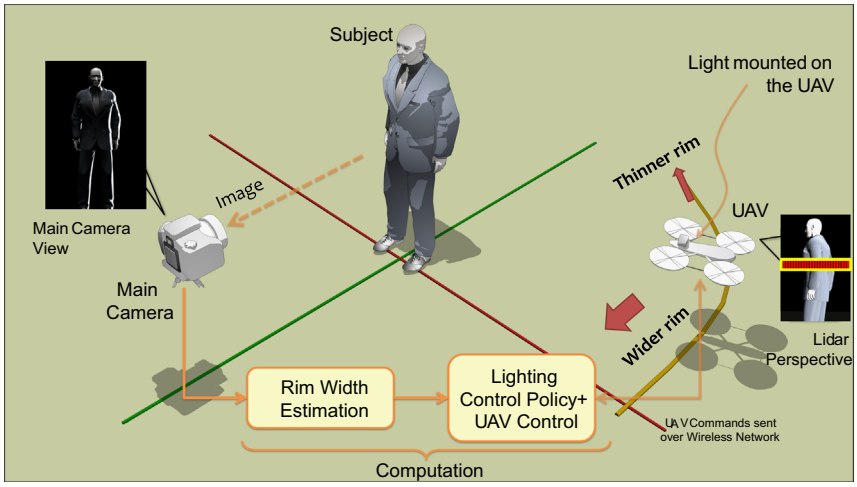

Fig. 2. *System overview*: the photographer shoots the subject with the main camera; an aerial robot, a quadrotor, equipped with a light source illuminates the subject from the side as shown. The video images from the main camera are analyzed to evaluate the quality of rim light and consequently, the robot position is continuously adjusted to achieve a desired quality of rim light effect. Positioning the robot closer to the side of the subject produces a "wider" rim light effect, while positioning it behind the subject leads to "thinner" rim light effect.

capture system. Our motion capture system uses retroreflectors that are mounted on the quadrotor. Using an array of cameras, the motion capture system triangulates the position and orientation of the quadrotor.

4. Rim width computation

Our input is an image where rim lighting dominates and we know on which side of the subject the light is (right vs. left). We identify pixels that correspond to the subject boundary that is strongly illuminated from one side and compute the width and orientation of the silhouette. Given a distribution of orientations and rim widths, we focus on large clusters or peaks as they usually correspond to areas of the image with well-defined consistent rim lighting. We use the average width of these large rim areas as our global characterization of rim lighting. While the average rim width across the image does not claim to directly measure the aesthetic quality of rim lighting, it provides a reliable control for the photographer who can increase or decrease the desired width. Our solution is based on simple image processing and other rim width characterizations could be used as well, as long as they yield stable and predictable control of the robot.

4.1. Computing rim width and orientation

We are interested in areas of the image where rim lighting is prevalent and has a consistent width that emphasizes silhouettes. For this, we first make sure that the onboard light dominates and we slightly overexpose the image so that bright pixels correspond to rim light.

Silhouette pixels: To identify silhouette pixels, we build on ideas from Harris corner detection except that we seek to identify edges and exclude corners because rim width is poorly defined for them. We also wish to focus on vertical edges because our main goal is to control the horizontal location of the aerial robot, which mostly affects rim width at vertical silhouettes. Finally, rim lighting usually locally starts at the occluding contour of the subject and stops where light is at the grazing angle and the cosine term of irradiance goes to zero. We want to extract pixels corresponding to the former (silhouettes) and use the sign of the dot product between the rough 2D light direction and the image gradient.

Putting all these objectives together, we define a light-direction biased structure tensor operator $\mathcal{G}[\cdot]$ as

$$\mathcal{G}[\cdot] = \begin{bmatrix} \left(\mathcal{L}\frac{d}{dx}\cdot\right)^2 & \left(\mathcal{L}\frac{d}{dx}\cdot\right)\left(\mathcal{L}\frac{d}{dy}\cdot\right) \\ \left(\mathcal{L}\frac{d}{dy}\cdot\right)\left(\mathcal{L}\frac{d}{dx}\cdot\right) & \left(\mathcal{L}\frac{d}{dy}\cdot\right)^2 \end{bmatrix} \tag{1}$$

where the light-direction bias operator $\mathcal{L}[\cdot]$ emphasizes pixel gradients that are horizontal and towards the light and is defined as

$$\mathcal{L}[dv] = \begin{cases} dv \cdot L_v & \text{if } (dv \cdot L_v > 0) \\ 0 & \text{elsewhere} \end{cases} \tag{2}$$

where dv is the gradient vector at the pixel and L_v is the unit vector at the pixel pointing towards the light source projected on the image plane. We use Eq. (2) to filter out gradients that are not due to the rim light source, for instance shadows and surface variations.

Let I_M be the input intensity image and $\mathcal{F}[\cdot]$ a low-pass filter operator. We compute the structure tensor of the input image I_G by low-pass filtering both input and output as

$$I_G = \mathcal{F}[\mathcal{G}[\mathcal{F}[I_M]]] \tag{3}$$

Using eigen decomposition of I_G, we obtain eigenvalues and vectors as

$$I_G(i,j) = U\Lambda U^{-1} \tag{4}$$

where the diagonal entries of Λ contains the two eigenvalues for pixel (i,j), and U contains the two eigenvectors. For each pixel, we can determine if it is a strong edge and not a corner by looking at the difference in eigenvalues. This gives us I_F, which measures the strength of the silhouette edge at each pixel

$$I_F = \frac{(\Lambda_{1,1} - \Lambda_{2,2})}{\sqrt{(\Lambda_{1,1} + \Lambda_{2,2}) + \epsilon}} \tag{5}$$

The orientation angle of the edge is given by

$$I_A = \tan^{-1}\left(\frac{U_{1,1} - \Lambda_{2,2}}{U_{2,1}}\right) \tag{6}$$

Note that, we arrange Λ and U such that $\Lambda_{1,1} \geq \Lambda_{2,2}$. ϵ is a small positive quantity to avoid divide-by-zero.

Local width: Using I_F and I_A, we compute the rim width for all strong edge pixels, that is for pixels $I_F(i,j) > \text{Threshold}$. We construct a line segment $L_{i,j}$ that starts at pixel (i,j) and is directed at an angle given by $I_A(i,j)$. We sample the input image I_M along $L_{i,j}$ to obtain a 1D array of pixels. The 1D array contains a pulse that starts with high intensity and eventually falls off. We estimate the width of the pulse, which is the rim width *rw* corresponding to the pixel (i,j). While computing *rw*, we start scanning from (i,j) and continue to count the pixels until we reach a pixel value that is a certain threshold lower than the peak value encountered. The total pixels counted correspond to pulse width, or the rim width. We also store the rim orientation $ro(i,j) = I_A(i,j)$ for all pixels such that $I_F(i,j) > \text{Threshold}$. Fig. 3 shows the rim widths and rim orientations for an input image.

4.2. Rim peaks and average width

The above method provides us with a distribution of rim widths and orientation and we now need to aggregate it into a single control variable that characterizes the overall rim lighting in the photo.

We observe that rim lighting is visually most salient in areas of a picture where rim width is consistent both in orientation and size. We use a simple heuristic to focus on such rim light: we extract the peaks of a 2D histogram along orientation and width (Fig. 3, right). That is, using the rim width *rw* and the rim orientation *ro*, we construct a 2D histogram where each bin encodes the number of pixels with a particular width and orientation. In our implementation we used 180 bins for the orientation angle in degrees and 80 bins for the widths. We identify local peaks in the histogram by first thresholding and then computing local maxima.

Finally, we compute the average width of these peaks, which provides us with our control variable $rw_{average}$. We also use the width variance $rw_{variance}$ to diagnose cases where some regions of the image have excessively large rim width compared to the average (such as the left of Fig. 4), in which case we seek to reduce the average (see Section 5.1).

5. Control

Our control strategy is motivated by a number of goals, which naturally map to the degrees of freedom of the aerial robot (Figs. 5 and 6), and require corresponding sensor inputs. (1) We focus on the control of the robot along a horizontal plane. (2) Our main objective is to achieve a given rim width using sensory input from the photographer's camera, which can be achieved by controlling the polar coordinate ψ_{QS} of the robot around the subject. (3) We need to make sure that the light carried by the robot is directed towards the subject, inducing a desired yaw orientation, and assessed using a robot mounted lidar. (4) The robot must remain at a given safe distance from the subject, which is also measured by the lidar. The remaining degrees of freedom are controlled by the required constant height and the internal dynamics of the robot. In this section we use the terminology of *Q* and *S* for the quadrotor and subject respectively.

Efficient control of the delicate robot dynamics demands rich sensor inputs, that is measurement of positions, their rates and so on. Although the robot boasts a variety of sophisticated onboard sensors (gyros, accelerometers, tilt sensors, height sensors, etc.), additional sensing becomes essential to ensure robustness and fast response. Because some of the onboard sensors have limited temporal rate, latency, and accuracy, we enrich these sensory data with externally located sensors (an indoor GPS). Although this may hinder portability, we trust the future technological progress in the field of state estimation will alleviate the need of external sensing. Moreover, we limited ourselves to available robot hardware with a given set of sensors.

We use a standard hierarchical control approach where our outer layer is used to control two levels of inner layers that run at increasingly higher rates and rely respectively on data about the 3D localization of the robot. Our contribution in this paper pertains to how we organize various degrees of freedom into control layers based on functional requirement and sensory data quality. This paper focuses on the outer layer, however we also cover the inner layers for completeness. We first cover the dynamics of the quadrotor aerial robot before describing its control. For a greater exposition on the dynamics and control of a quadrotor, readers may refer to some of the earlier works of Lozano et al. [19] and Bouabdallah et al. [20].

5.1. Lighting control

We first focus on the control of the quadrotor location along the horizontal plane. We will discuss other degrees of freedom in the following sections. Our objectives are best expressed in polar coordinates around the subject (Fig. 5). Let ψ_{QS} be the angle in the 2D plane between the quadrotor and the world *x*-axis. ψ_{QS} affects the rim width, however we also want to maintain d_{safe} distance to the subject to ensure safety as well as constant light intensity.

Angle: Given the current average rim width computed from the main camera (Section 4), we need to move the aerial robot around the subject to achieve a specified width. We cannot directly compute the optimal ψ_{QS} and we only know the directions that will increase and decrease the width. This is why we use an iterative process that seeks to modify the polar angle ψ_{QS} at a speed proportional to the mismatch of rim width. At each iteration of the outer control loop, we

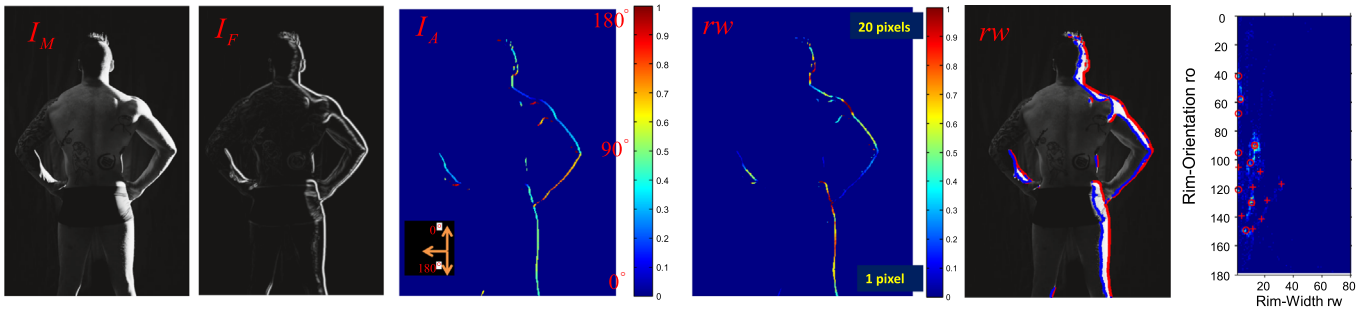

Fig. 3. *Left to right*: (1) input image I_M, (2) boundary pixels I_F, (3) rim orientation angle I_A, (4) rim widths *rw* marked on the input image I_M as color coded pixels, (5) rim widths *rw* marked as interior (blue) and exterior (red) points on the input image, and (6) the 2D histogram with peaks highlighted by marker. In (6) the highest peaks are marked as squares, the intermediate peaks are marked as circles, and smaller peaks are marked by +.

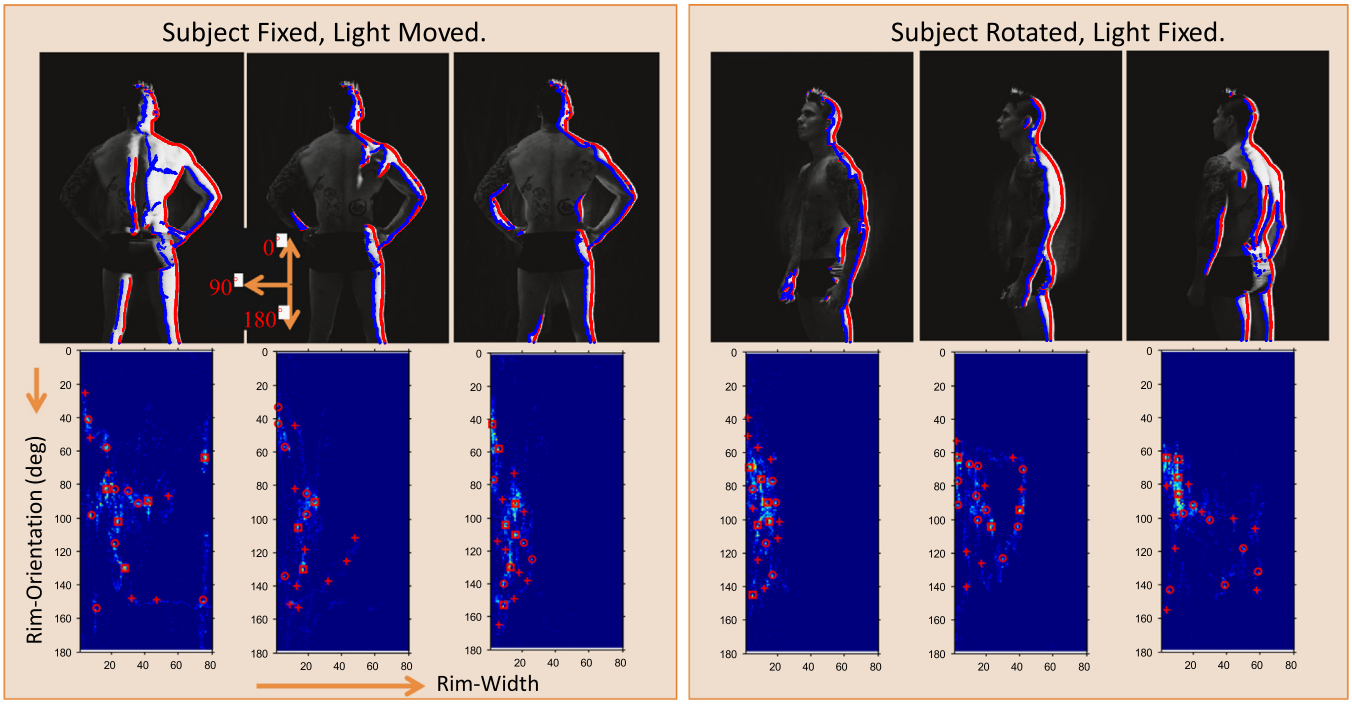

Fig. 4. *Left-box*: a set of input images I_M with varying light location and fixed subject posture. From left to right, the light moves from the side to the side-rear. The corresponding histogram is plotted below. The Histogram is overlaid with markers at peak locations – a square marker indicate peaks greater than 75%, circled markers indicate peaks between 50% and 75% and + markers indicate peaks between 25% and 50%. We notice that as the light moves to the rear, the rim width becomes smaller and the peaks in the histogram cluster towards the left hand side. *Right-box*: a set of input images I_M with varying subject posture and fixed light location. From left to right, the subject rotates, exposing different body curvatures, resulting in varying rim widths. The corresponding histogram is plotted below the image. We notice that a change in subject posture can dramatically change rim widths.

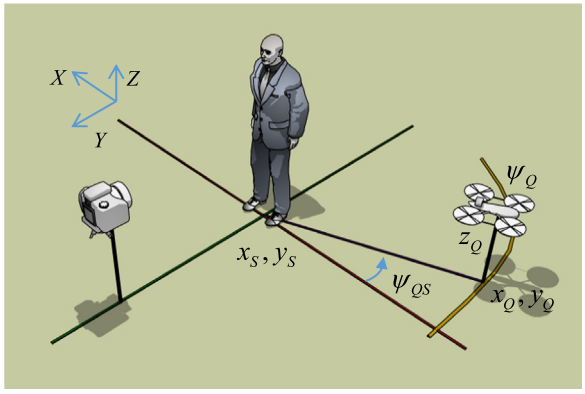

Fig. 5. Coordinate system.

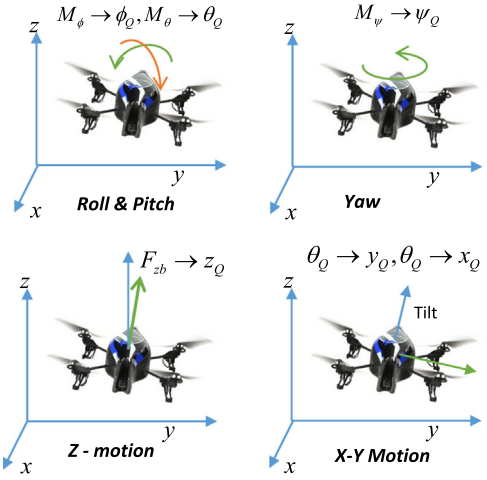

Fig. 6. Quadrotor motion as a result of various control inputs.

compute the desired rate of angle change as

$$\Delta\psi_{QS,des}(t) = k(rw_{average}(t) - rw_{desired}) \tag{7}$$

where $rw_{average}(t)$ is the average rim width at time t. $rw_{desired}$ is the desired rim width and k is a constant for numerical normalization. In addition, we use dead zoning, where we zero out the difference if it is within a small threshold (usually set at 3 pixels in our implementation) to fall back to stable hovering when the width is close enough.

If the variance of rim width is above a threshold (15 in practice), we also want to move the UAV to reduce rim width and use

$$\Delta\psi_{QS,des}(t) = k(rw_{average}(t) - rw_{desired}) + k_2(rw_{variance}(t)) \tag{8}$$

Given this desired rate of angular change $\Delta\psi_{QS,des}$, we implement the following control law to generate the desired angle $\psi_{QS,des}$ with an integral controller:

$$\psi_{QS,des} = \int_0^t K_{adapt}\Delta\psi_{QS,des}\, dt \tag{9}$$

where K_{adapt} is the adaptation gain that influences how fast or slow the quadrotor moves in response to the input $\Delta\psi_{QS,des}$. We add limits on the integrator that essentially restricts the quadrotor from entering the field of view of the camera as it goes around the subject. Large K_{adapt} means the quadrotor adapts quickly, but at the expense of overshooting and ringing, which is a natural consequence of dynamics. While this issue may be a problem for continuous adaptation, we used this as an optional feature. Instead of allowing the quadrotor to settle to its optimal point, we trigger the camera automatically when the quadrotor passes the optimal point, and then use the post capture time (when the photographer

is reviewing the photograph) to allow the quadrotor to settle in the optimal position.

Distance and location: Using the lidar, we estimate the position of the subject in the quadrotor reference frame and translate it to the world frame to obtain x_S, y_S. The algorithm for subject detection is based on template matching, a standard approach. Our goal now is to achieve and maintain a fixed distance d_{safe} from the subject. Along with the subject position, the desired angle $\psi_{QS,des}$ and d_{safe} provide us with the next desired 2D position of the robot $x_{Q,des}, y_{Q,des}$:

$$\begin{bmatrix} x_{Q,des} \\ y_{Q,des} \end{bmatrix} = \begin{bmatrix} x_S + d_{safe}\cos(\psi_{QS,des}) \\ y_S + d_{safe}\sin(\psi_{QS,des}) \end{bmatrix} \tag{10}$$

In the event of subject position not being detected, previously known position of the subject is used (dead reckoning). To avoid collision with any object in general, an internal guard band violation from lidar triggers an emergency landing.

5.2. Low-level control

Each iteration of the outer loop provides a desired 2D location. With the addition of the specified height and the yaw that aligns the quadrotor with the subject, this specifies the four actuated degrees of freedom of the quadrotor. The intermediate loop uses data about the current localization and orientation of the quadrotor (provided by the motion capture system and the onboard sensors) to guide the robot towards this 3D point and orientation. This is a standard robotics task, albeit challenging because the system has unstable dynamics and is under actuated. We include our control strategy below for completeness but it mostly follows standard robotics approaches. The reader who is not interested in low level control issues can directly skip to Section 5.3.

Simplified dynamics of a quadrotor: A quadrotor consists of a rigid frame with four propellers (actuators) that in combination produce a thrust force F_{zb} along its body z-axis, and three body torques (rotation forces) M_ϕ, M_θ, M_ψ. These forces are electronically controlled in order to make the quadrotor follow a desired motion, or hover at a fixed 3D location. Since the quadrotor has six degrees of freedom, that is, position x_Q, y_Q, z_Q and Euler rotation angles ϕ_Q, θ_Q, ψ_Q, and only four force inputs, we can control only four of the six degrees simultaneously. This leads to a well-known problem of underactuatedness, thus demanding a challenging control design. The control inputs M_ϕ, M_θ, M_ψ directly influence respectively the angles ϕ_Q, θ_Q, ψ_Q, also known as roll, pitch and yaw. This is illustrated in Fig. 6. There is no direct control of $x-y$ motion from the input, instead, this is achieved by controlling the internal variables ϕ_Q and θ_Q.

The approximate rotational dynamics of the quadrotor can be summarized as

$$\begin{bmatrix} \ddot{\phi} \\ \ddot{\theta} \\ \ddot{\psi} \end{bmatrix} \approx \begin{bmatrix} \frac{1}{J_x} M_\phi \\ \frac{1}{J_y} M_\theta \\ \frac{1}{J_z} M_\psi \end{bmatrix} \tag{11}$$

where $J_{x,y,z}$ is the moment of inertia of the quadrotor. The dynamics along the z-axis is given by

$$\ddot{z}_Q \approx -g + (\cos\phi_Q \cos\theta_Q)\frac{F_{zb}}{m_Q} \tag{12}$$

where g is the acceleration due to gravity, and m_Q is the mass of the quadrotor. As mentioned earlier the horizontal $x-y$ dynamics depend on the rotational dynamics and not directly on the control input, and this can be written as

$$\begin{bmatrix} \ddot{x}_Q \\ \ddot{y}_Q \end{bmatrix} \approx \begin{bmatrix} \cos\psi_Q & -\sin\psi_Q \\ \sin\psi_Q & \cos\psi_Q \end{bmatrix} \begin{bmatrix} g\theta_Q \\ -g\phi_Q \end{bmatrix} \tag{13}$$

Intermediate loop control systems: Here we describe the control of $x-y$ position of the quadrotor with reference to the subject. Given Eq. (13), we control ϕ_Q and θ_Q in order to control x_Q and y_Q as follows:

$$\begin{bmatrix} \theta_{Q,des} \\ \phi_{Q,des} \end{bmatrix} = \begin{bmatrix} \cos\psi_Q & \sin\psi_Q \\ -\sin\psi_Q & \cos\psi_Q \end{bmatrix} \begin{bmatrix} C_{PID}(x_{Q,des} - x_Q) \\ C_{PID}(y_{Q,des} - y_Q) \end{bmatrix} \tag{14}$$

Inner loop control systems: The structure of the inner loop rotation controllers is given by

$$\begin{bmatrix} M_\phi \\ M_\theta \\ M_\psi \end{bmatrix} = \begin{bmatrix} C_{PID}(\phi_{Q,des} - \phi_Q) \\ C_{PID}(\theta_{Q,des} - \theta_Q) \\ C_{PID}(\psi_{Q,des} - \psi_Q) \end{bmatrix} \tag{15}$$

where, the proportional-integral-derivative (PID) controller is defined by

$$C_{PID}(e) = eK_P + K_I \int_{-\infty}^{t} e\,dt + \dot{e}K_D \tag{16}$$

where K_P, K_I, K_D are control gains that indicate how the error e influences the correction forces. The subscript *des* denotes the desired quantity that we want the quadrotor to take. Except for ψ_Q, all measurements are made with onboard sensors. The height of the quadrotor is controlled using

$$F_{zb} = \frac{m_Q g + C_{PID}(z_{Q,des} - z_Q)}{\cos\theta_Q \cos\phi_Q} \tag{17}$$

The height measurement z_Q is usually done using the ultrasonic distance sensor on the quadrotor. In the event of unreliable height sensing, external measurements of z_Q are used.

5.3. Aggressive control mode

We implemented two modes of operation: (1) continuous mode and (2) aggressive, autotrigger mode. In a continuous mode, the quadrotor continuously adapts to the changes in subject posture and viewpoint. When the subject makes sudden posture changes, the photographer has to wait until the quadrotor moves to the new location (which may be on the order of 1–2 s in extreme cases) and then take a photograph. In mode (2), the photographer presses the shutter button half way through (a standard feature in professional cameras that starts auto-focusing, etc.), which we recognize and use to initiate the quadrotor motion. In this case we set the gain factor, K_{adapt} in Eq. (9), to a much higher value which causes the quadrotor to move very fast but over shoot. The instant when the quadrotor passes through the optimal location, we automatically trigger the camera capturing the photograph. We then allow the quadrotor to over shoot, slow down, and settle to the optimal point. The hardware we developed to sense the half-press of the shutter button, and to trigger the camera involves a microcontroller that communicates with the *remote-trigger* port of the camera.

6. Results

We demonstrate our approach on various subjects including a plastic mannequin, and male and female models with varied clothing. We show photographic results, as well as experimental plots to validate our approach.

6.1. Performance

The rim computation was performed at ~23 fps while the quadrotor onboard inner loop controller ran at around ~200 fps. The intermediate loop ran at around ~60–100 fps and the lidar provided depth scans at ~10 fps. The height of the quadrotor was chosen to be between 750 mm and 1200 mm, depending on the photographer's choice.

6.2. Photographic results

Fig. 1 (top row) shows photographs captured for various desired rim width $rw_{desired}$ settings. Fig. 7 shows results for the same $rw_{desired}$, where the aerial robot achieved the desired width within 2–3 pixel tolerance.

Fig. 8 shows a female subject in sitting position, performing a yoga posture. In the second and the third image, $rw_{desired}$ is the same but we notice that a change in posture has made the quadrotor shift to a new location. A female subject in Fig. 9 performs several postures, both in upright and sitting position.

Fig. 10 shows a subject with clothing that is not tight against their body. The second photograph shows the subject with a large camera lens. Because the lens has a cylindrical geometry with varying radii, the histogram has few strong peaks. In this case, the photographer specified the desired peak location for the rim width optimization.

Fig. 7. Photographic results optimized for average rim width. For $rw_{desired} = 9$, the lighting optimization resulted in $rw_{average} = 7.2$ in the case of the left image, and $rw_{average} = 11.5$ in the case of the right image.

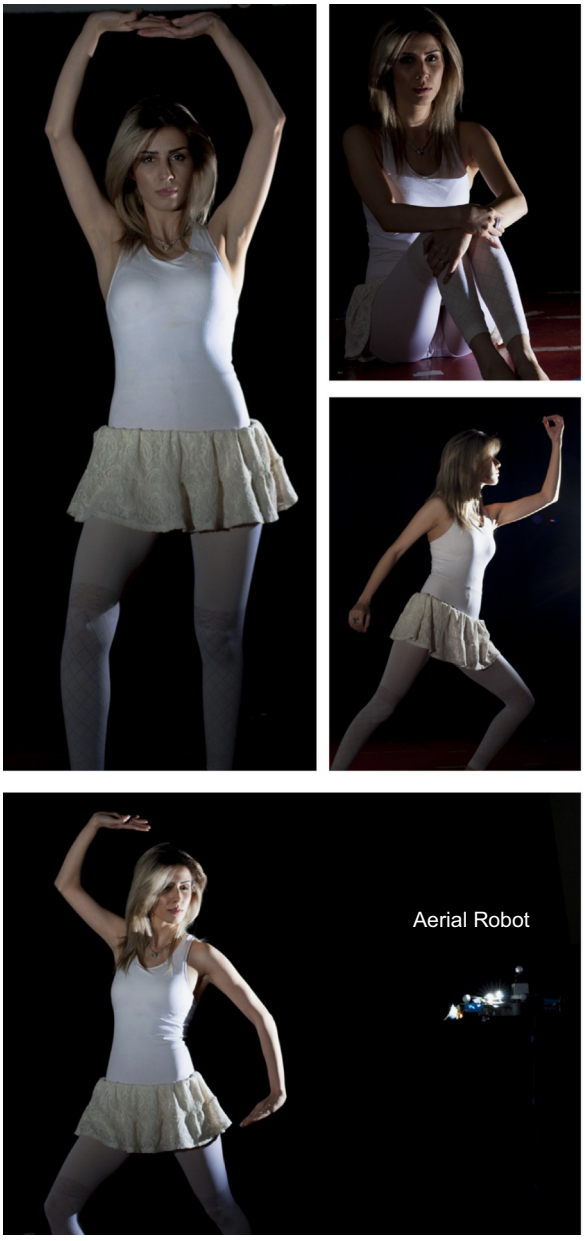

Fig. 9. A female model performs several ballet poses. The photographer chose to have the same $rw_{desired} = 10$ for all the above images. A fill light was also triggered during the capture process. The image on the top left was captured in aggressive, auto trigger mode, while the rest were captured in continuous adaptation mode.

Fig. 8. *Left*: a subject performing yoga postures. When the subject changed in the posture from what is shown in the center image to what is shown in the right image, the quadrotor moved to a new optimal location, traveling about a meter. In the center image, the rim on the torso dominated and hence the quadrotor had to move back to reduce the average width. However, in the right image, where the legs are upward folded, thinner rims now caused the average rim width to drop, as a result, the quadrotor moved forward. Notice how the torso is over lit in the right image. This is a consequence of wide range of curvatures in the subject.

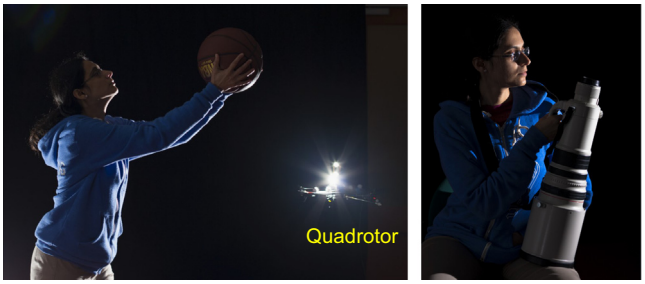

Fig. 10. *Left*: subject with clothing that is not tight. *Right*: the subject holding a lens, which cased a strong dominant rim by setting $rw_{desired} = 20$, with actual achieved $rw_{average} = 17$.

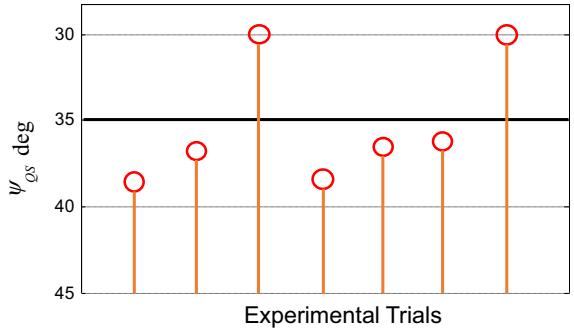

Fig. 11. Starting from the same initial location of $\psi_{QS} = 60^\circ$, the quadrotor reached roughly to the same spot for several trials. The camera and subject remained fixed. The mean is computed to be 35° and variance 12. In each case the quadrotor moved to the optimal point in about 2.4 s, with an average speed of 30°/s.

6.3. Technical evaluation

Evaluation of repeatability: We fix the camera and the mannequin subject, and $rw_{desired}$, and conduct several experimental trials starting the quadrotor (hovering at fixed height) with the same initial position of $\psi_{QS} = 60^\circ$. We then start the optimization process and record the angle ψ_{QS} when the quadrotor reached the optimal location, that is when $rw_{desired} \approx rw_{average}$. Fig. 11 shows the final angles achieved. We observe that the robot consistently reached the same location within a tolerance of $\pm 5^\circ$.

Evaluation of continuous adaptation: In this experiment, we fixed the camera and allowed the quadrotor to continuously adapt to the changing posture of the subject. We rotated the mannequin by minimally interfering with the scene. We evaluated two types of motion, slow and jerky, as indicated in Fig. 12. For this experiment, the photographer indicated that a tolerance of ± 3 pixels was acceptable.

With the jerky motion, the quadrotor quickly returns to the optimal location in about 2 s. For small motions of the mannequin, the quadrotor tracked the change, and adapted its location to maintain a constant rim width (within the specified tolerance range).

Adaptation to changes in viewpoint: In the following experiment, a female model posed while the photographer moved between positions C_1 and C_2 as shown in Fig. 13. The change in viewpoint caused a change in lighting effect as seen from the camera. The robot responded to this change automatically and continuously moved from Q_1 to Q_2 in order to maintain the same desired average rim width. This result demonstrates our system's capability to adapt to motion of the photographer.

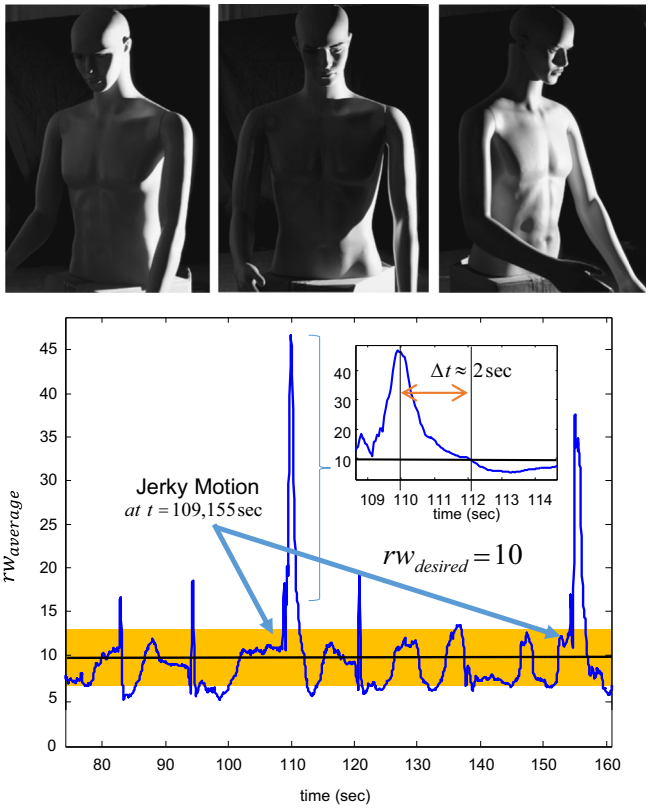

Fig. 12. Plot of rim width $rw_{average}$ vs. time as the mannequin is rotated about its center. With $rw_{desired} = 10$, we note that the quadrotor constantly tried to bring the $rw_{average}$ towards $rw_{desired} = 10$. In the extreme cases when the mannequin was rotated by over 45° with a jerky motion at $t = 109$ s, indicated by a strong peak in the plot, the quadrotor moved to the optimal location within about 2 s. In cases when the mannequin was moved gently, the quadrotor kept pace and constantly adapted. The small deviations from $rw_{desired} = 10$ can be attributed to both the specified tolerance, as indicated by the yellow band, and the quadrotor's noisy motions. The top row of images (not rim width optimized) show some of positions of the mannequin during the experiment.

6.4. Monotonicity of rim width variation

In our control scheme, our strategy was to move the light backwards until a desired rim width is achieved. This is based on the assumption that moving the light backwards reduces rim width, ideally in a monotonic sense. The validity of this assumption is hard to prove mathematically since the rim width (as we define it) depends on factors such as geometry of the subject, position of camera and the light, and of course the reflective properties of the subject. One way to empirically show the monotonic variation of rim width with respect to light position is to perform several experiments with a variety of subject geometries and materials. Results in Figs. 11 and 13 inherently justify this assumption. However, for more accurate insights, we performed simulations with simple 3D objects within a rendering system. Fig. 14 shows a rendering setup where a virtual camera is setup to view the object while a point source light is moved from position Q_1 to Q_2. Rim width is computed for each position Q_i and plotted against the position Q_i. If the object were a cylinder with diffuse surface, it is a trivial process to show monotonicity. We considered a couple of interesting cases. In the first experiment of Fig. 14, we used a rectangular cube such that the flat surface faces the camera. As the light moves to the side, at a certain point the light becomes invisible to the front surface and hence the rim width vanishes to zero. This is an interesting case that highlights the abrupt discontinuities in rim width, nevertheless it is still a monotonic behavior. Abrupt jumps in rim width can negatively impact the robot control system since it may introduce unwanted oscillations (referred to as limit cycles). For instance, if the photographer specifies a desired rim width of 20 pixels, there is no light position that can possibly achieve this, the robot simply

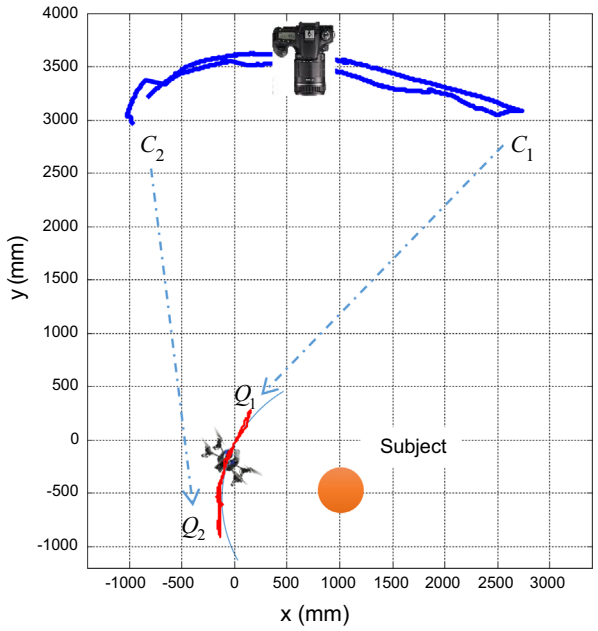

Fig. 13. The top view shows the $x-y$ trajectory of the camera and the robot. A female subject performed gentle posture changes while the photographer moved between positions C_1 and C_2, shown in the blue trajectory. To maintain a consistent average rim width, the robot moved between Q_1 and Q_2, shown in red. The circle represents the subject's location. All distances are in mm.

keeps moving back and forth. In the second case, we used a more complex object like the teapot (with diffuse surface). In this case, the light is casting self-shadow on to the object. Sharp shadows can wrongly interfere with rim width computation to produce false average rim width. This leads to non-monotonic behavior as shown in the plot. Thanks to our rim width algorithm, we discount for inconsistent edges, thus the effect of shadows is not severe and the behavior is still approximately monotonic. We also performed an experiment where we disabled shadow computation (not shown in figure) and noticed that monotonicity was maintained.

6.5. Rim width compositing

While capturing the final photograph (where the light position is optimal) is one objective of our work, we can also combine multiple photographs captured with various light positions. For complex geometries with a wide range of curvatures, any given light position may not achieve consistent rim widths for all parts of the object. Desirable rim widths on thinner parts may lead to over exposed thicker rim width on larger parts; and thin rim widths on larger parts would mean that the thin parts do not receive light at all. An alternative is to capture multiple images such that each part of the subject has its own optimized light position. We can then produce a composites image to produce a consistent rim width on the whole subject. We address this issue in the case of static or near-static subjects. We prototyped an approach that takes a number of images with varying light locations (that is, for various ψ_{QS}) and composites them by selecting regions from each image that have a satisfactory rim widths (Fig. 15).

The algorithm to composite multiple images is as follows. For each image, we evaluate the rim widths for all pixels as described in Section 4. Given the desired rim width, we generate a mask-image that has high weights for pixels that have the desired rim width and low weights otherwise. We use the difference in desired rim width and the measured rim width as a way to compute the weight. That is, weight $= e^{-k(rw_{desired}-rw)^2}$, where k is the fall off factor. We ensure that

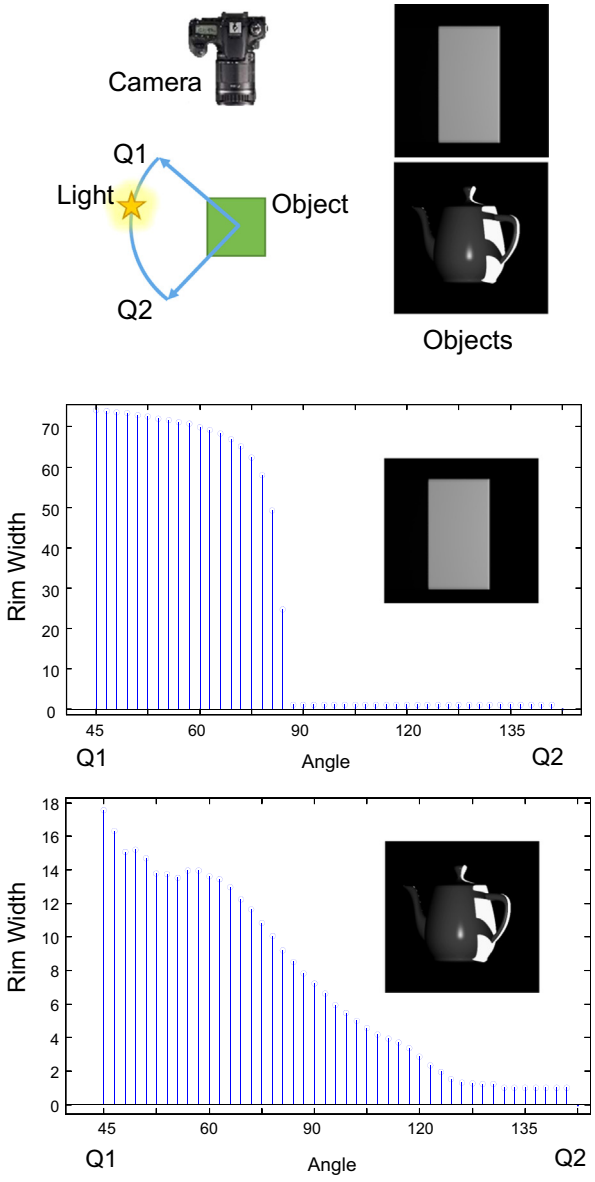

Fig. 14. Simulations to show approximate monotonic behavior of rim width with respect to light position.

at each pixel, the mask-image sums to 1 across all images by normalization. Using the mask-images as the blending weights, we combine all the images to produce the final composite. The blending function is such that the pixel with highest weight wins. However, to reduce artifacts, we take contributions from second highest and the third highest weights to contribute to the final composite. For the mannequin result shown in Fig. 15 we captured still images at 14 fps; the flash was hand held, and was synchronously triggered at 14 fps. For the case when the subject is not fully rigid or when there is slight motion in the camera, the images have to be co-registered before compositing. We found that patch-match [21] or non-rigid dense correspondence algorithms [22] can be employed to co-register images with slight movement. However, such correspondence algorithms work best when the scene lighting is similar across images. Therefore, more dense sampling of light positions is desirable.

6.6. Outdoor experiments and discussion

In this section, we present an open-loop outdoor robotic lighting setup. While work reported by Srikanth et al. [3] shows the efficacy of

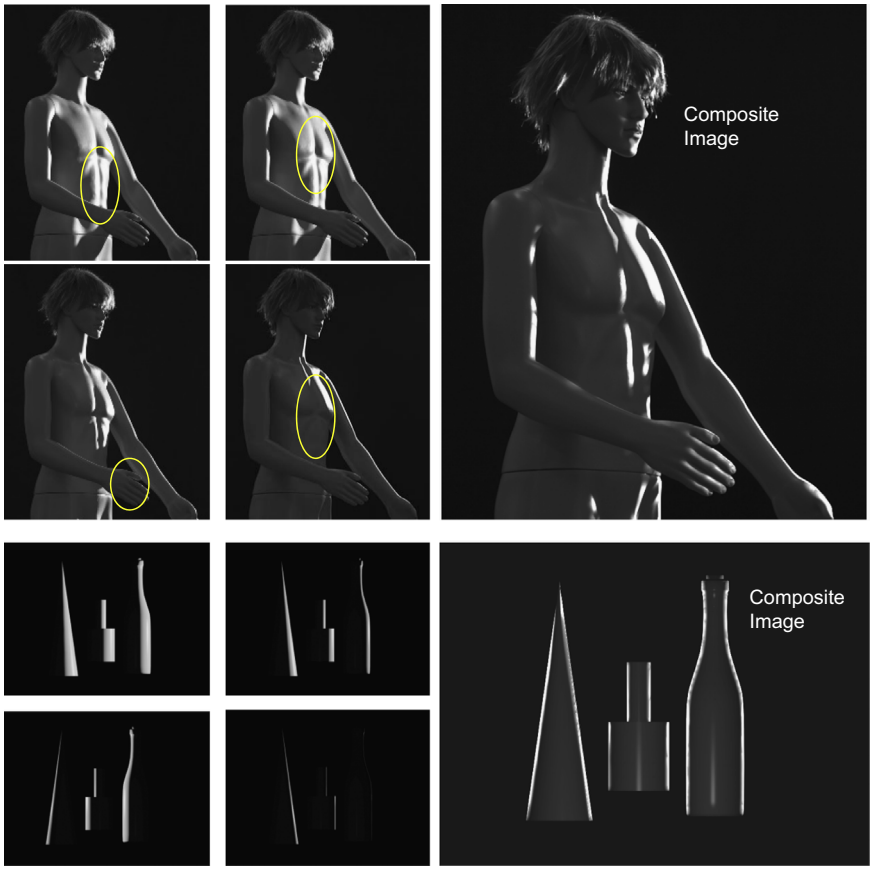

Fig. 15. Smaller images on the left are a subset of images captured with varying light position. Notice the variation of rim widths because of the wide range of curvatures within a given input image. The yellow ellipses show the regions of the image that either have excessive or no rim at all. The larger image on the top right is an automatically composited rim image using around 20 images. The image itself was captured using a hand-held flash. The bottom example is a rendered scene where the light was moved in full circle around the three objects.

robotic lighting, its performance in terms of light power (lumens capacity of flash) and the flight time were limited. Due to recent progress in aerial robotics technology, a number of new platforms with far more weight carrying capacity and flight-time have become available. For our outdoor studies, we used a DJI Phantom aerial robot [23]. We attached a standard camera flash which is approximately 20× more powerful than the one used by Srikanth et al. [3]. Our outdoor system does not contain onboard Lidar to recognize and track the person, hence it is an open loop system. The main intent of this experiment is to study the lighting ergonomics from the photographer's perspective and to understand the capabilities and limitations of the robot in a realistic photo-shooting scenario. For instance, to study the stability of the robot during a photoshoot while carrying a standard flash. We also wanted to study how a photographer would go about using a robotic system since it is a different system in comparison to a lighting assistant.

Fig. 16 shows the lighting robot hovering about 10–15 ft away from the subject, and still able to produce a significant impact in terms of lighting. For reference, photographs without lighting is presented in the lower row. Note that the lighting is performed when there was significant amount of ambient light. The examples on the right column shows the robot flying next to a foot-bridge about 20 ft high, where conventional lighting would have been impossible. We recorded a continuous flight time of about 25 min. The flash itself is capable of firing more than 400 full-powered light bursts before requiring change of batteries. In our case, the duration for each continuous photo session lasted for at about 8–10 min of shooting. Therefore, the robot only needs to be in the air for about 10 min a stretch, which leaves ample time to swap batteries. In an outdoor setting, the noise from the robot gets dissipated easily and the wind disturbance is negligible (ambient wind flow is more pronounced than the wind from the robot itself).

The typical flow in our outdoor lighting scenario is as follows. The photographer first decides the type of lighting style. The approximate position and postures for the subject are decided. The photographer decides his/her own approximate shooting position. Using necessary lenses and camera, the photographer starts to take trial images, progressively adjusting the light intensity and positions of the lights. In our case, we used one fill light, which is on the ground and close to the photographer, and one robotic light which is hovering close to the subject. The fill light's position is not critical, and hence barely requires position adjustments. Whereas the position of the robotic light (which in our case is responsible for the specific style) is more critical. The photographer then adjusts (in our cases, instructs the person controlling the robot) the robot's position relative to its previous position, successively moving it backwards or forwards (and upwards or downwards). This process is mostly local and the adjustments are mostly relative.

The key insight is that there is only a limited degree of freedom left for lighting once the photographer (or the director) decides on a specific theme. Even so, significant effort goes into optimization within the restricted degree of freedom in the case of conventional lighting (that is, assistant moving the lights). Developing objective functions to find optimal lighting positions within this restricted space is more tractable than solving a generic global lighting optimization problem. This is because there is a good chance that a well-defined optimum exists within the restricted space, and that searching for one may be relatively straightforward. This also goes

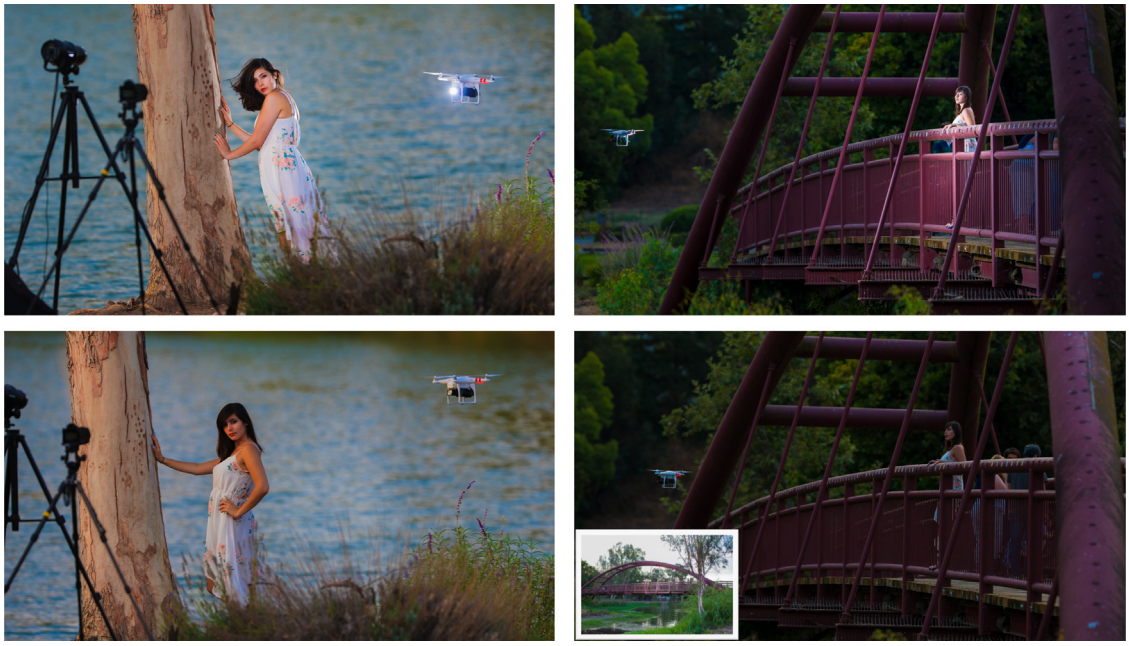

Fig. 16. Aerial robotic lighting in outdoor setting. Top row shows photographs taken with robotic lighting and the bottom rows show without lighting. The right column shows the robot lighting the subject on a bridge, which is impossible with conventional lighting just because of the accessibility issues. The inset on the bottom right shows the wider view of the scene.

well with the fact that photographers desire to have complete control over the lighting style, but are open to accept last step optimization performed by the computer since it saves significant amount of time, and increases the chances of getting good shots. In our case, we used rim lighting optimization where the initial *xy* position and the height of the robot is decided by the photographer. The computer only optimizes in the limited space of approximately one quadrant.

When the subject reposes or moves, our strategy is to maintain a consistent lighting with reference to the previously optimized setting. That is, we maintain the same rim width when the subject reposes. If the subject significantly reposes, the new optimal light position computed by the algorithm, even though it satisfies the objective function, may not be satisfactory to the photographer. However, given that there is a high chance the photographer may miss the shot due subject motion, obtaining a sub-optimally lit photograph is better than not capturing any.

6.6.1. Further discussions

We have demonstrated a proof-of-concept system that can achieve a desired rim effect but there are several potential improvements.

Energy efficiency: While we used a continuous light source, a more energy efficient approach is to use pulses of light that are synchronized with the camera. This may save a significant amount of energy, which is a vital resource on an aerial robot.

Light modifiers: It would be worthwhile to attach suitable light modifiers that are lightweight, and that can aerodynamically conform to the aerial robot to enhance the quality of photographs.

Flash power: The size of the flash strobe is only limited by the payload capacity of the aerial robot. Although we used a very light and low cost aerial robot, a little additional cost can provide higher payload capacity. For instance, it is typical to have 0.5–1 kg of payload even with inexpensive quadrotors ($1000).

Outdoor usage: One issue of using our system outdoors is the presence of background. In order to evaluate the influence of our light on the subject, we need to remove the background. One possible way to suppress the background is to take two photographs in quick succession, one with a powerful flash strobe and another no-flash image, and then use the difference image to suppress the background.

Outdoor localization: In our setup, we used an indoor localization system to obtain some of the quadrotor measurements. In outdoors, by combining lidar data with global positioning data, and a magnetometer, one could still use the same control law. However, since these measurements are available at a low rate, the whole system may be slower. Another alternative is to setup a portable motion capture system which can be deployed outdoors with setup time comparable to the setup time of four portable studio lights.

Other platforms and lighting objectives: Our pipeline focused on rim lighting optimization and the use of a quadrotor. However, both ends of this pipeline can be a variable. For instance, one could use a Roomba-like wheeled robot to perform a similar function, perhaps with significantly higher payload capacity. One could also explore using aerial robots for other styles of lighting, for instance, back lighting and key lighting.

7. Conclusions

This paper presents a new approach to computational illumination where light sources are actuated and automatically adjusted to the view from the photographer's camera, which allows lighting to react in real time to changes in viewpoint, subject location, and pose. Our new rim lighting scheme handles dynamic subjects and relies on an aerial robot as a light carrying platform. We propose a new lighting control system based on the collaboration between the aerial robot and the main camera that achieves and maintains given rim lighting characteristics under changes in subject posture and position, and camera movement. Using an aerial robot, a quadrotor, and a few off-the-shelf components, we demonstrate our approach on a variety of subjects. Using an open-loop outdoor experimental setup, we show that our proposed strategies for lighting are inline with the ergonomics of the lighting. We believe that such a combination of computational photography, image analysis, and robotically actuated computational illumination can provide photographers

with greater flexibility in dynamic situations and dramatically reduce the need for a full crew of assistants.

Acknowledgments

We thank feedback from several lab members: Sylvain Paris, Vladimir Bychkovsky and YiChang-Shih for their help. We also want to thank the subjects appearing in the photographs: Joe Klamka, Nastaran Sharifi, Caithlin Palmer and Monali M. We want to thank photgrapher and drone flyers Barry Blanchard for his help.

Appendix A. Supplementary material

Supplementary data associated with this paper can be found in the online version at http://dx.doi.org/10.1016/j.cag.2015.03.007.

References

[1] Parrot. AR Drone Quadrotor, ⟨http://en.wikipedia.org/wiki/Parrot_AR.Drone⟩, [Online]; 2010.
[2] Pierre-Jean Bristeau, François Callou, David Vissière, Nicolas Petit, The navigation and control technology inside the AR. Drone Micro UAV, In: Proceedings of the 18th IFAC World Congress, Milano, Italy, 2011. p. 1477–1484.
[3] Srikanth M, Bala K, Durand F. Computational rim illumination with aerial robots. In: Proceedings of the workshop on computational aesthetics, CAe '14, ACM, New York, NY, USA; 2014. p. 57–66. doi:http://dx.doi.org/10.1145/2630099.2630105. URL ⟨http://doi.acm.org/10.1145/2630099.2630105⟩.
[4] Petschnigg G, Szeliski R, Agrawala M, Cohen MF, Hoppe H, Toyama K. Digital photography with flash and no-flash image pairs. ACM Trans Graph 2004;23 (3):664–72 URL ⟨http://doi.acm.org/10.1145/1015706.1015777⟩.
[5] Eisemann E, Durand F. Flash photography enhancement via intrinsic relighting. In: ACM transactions on graphics (TOG), vol. 23. ACM; 2004. p. 673–8.
[6] Raskar R, Tan K, Feris R, Yu J, Turk M. Non-photorealistic camera: depth edge detection and stylized rendering using multi-flash imaging. In: ACM transactions on graphics (TOG), vol. 23. ACM; 2004. p. 679–88.
[7] Debevec P, Wenger A, Tchou C, Gardner A, Waese J, Hawkins T. A lighting reproduction approach to live-action compositing, vol. 21. ACM; 2002.
[8] Anrys F, Dutré P, Willems Y. Image based lighting design. In: Fourth IASTED international conference on visualization, imaging, and image processing; 2004.
[9] Mohan A, Tumblin J, Bodenheimer B, Grimm C, Bailey R. Table-top computed lighting for practical digital photography. In: Rendering techniques 2005: 16th eurographics workshop on rendering; 2005. p. 165–72.
[10] Wang O, Fuchs M, Fuchs C, Davis J, Seidel H-P, Lensch H. A context-aware light source. In: IEEE international conference on computational photography. IEEE, 2010.
[11] Boyadzhiev I, Paris S, Bala K. User-assisted image compositing for photographic lighting. ACM Trans Graph 2013;32(4):36:1–12. http://dx.doi.org/10.1145/2461912.2461973 URL ⟨http://doi.acm.org/10.1145/2461912.2461973⟩.
[12] Dorsey J, Sillion F, Greenberg D. Design and simulation of opera lighting and projection effects. In: ACM SIGGRAPH computer graphics, vol. 25. ACM; 1991. p. 41–50.
[13] Dorsey J, Arvo J, Greenberg D. Interactive design of complex time dependent lighting. Comput Graph Appl IEEE 1995;15(2):26–36.
[14] Bousseau A, Chapoulie E, Ramamoorthi R, Agrawala M. Optimizing environment maps for material depiction. In: Computer graphics forum, vol. 30. Wiley Online Library; 2011. p. 1171–80.
[15] Pellacini F, Battaglia F, Morley R, Finkelstein A. Lighting with paint. ACM Trans Graph (TOG) 2007;26(2):9.
[16] Akers D, Losasso F, Klingner J, Agrawala M, Rick J, Hanrahan P. Conveying shape and features with image-based relighting. In: Proceedings of the 14th IEEE visualization 2003 (VIS'03). IEEE Computer Society; 2003. p. 46.
[17] Schoeneman C, Dorsey J, Smits B, Arvo J, Greenberg D. Painting with light. In: Proceedings of the 20th annual conference on computer graphics and interactive techniques. ACM, 1993. p. 143–6.
[18] Poulin P, Fournier A. Lights from highlights and shadows. In: Proceedings of the 1992 symposium on interactive 3D graphics. ACM; 1992. p. 31–8.
[19] Lozano R, Castillo P, Dzul A. Modeling and control of mini-flying machines, 2005.
[20] Bouabdallah S, Siegwart R. Full control of a quadrotor, in: IEEE/RSJ international conference on intelligent robots and systems, 2007. IROS 2007. IEEE; 2007. p. 153–8.
[21] Barnes C, Shechtman E, Goldman DB, Finkelstein A. The generalized patchmatch correspondence algorithm. In: Computer vision—ECCV 2010. Springer; 2010. p. 29–43.
[22] HaCohen Y, Shechtman E, Goldman DB, Lischinski D. Non-rigid dense correspondence with applications for image enhancement. In: ACM transactions on graphics (TOG), vol. 30. ACM; 2011. p. 70.
[23] DJI, DJI Phantom Quadrotor Drone, ⟨http://www.dji.com/product/phantom⟩, [Online]; 2014.