Proceedings of the Twenty-Ninth International
Florida Artificial Intelligence Research Society Conference

Selecting Vantage Points for an Autonomous Quadcopter Videographer

Rey Coaguila
Google
Mountain View, CA
reyc@google.com

Gita Sukthankar
University of Central Florida
Orlando, FL
gitars@eecs.ucf.edu

Rahul Sukthankar
Google
Mountain View, CA
sukthankar@google.com

Abstract

A good human videographer is adept at selecting the best vantage points from which to film a subject. The aim of our research is to create an autonomous quadcopter that is similarly skilled at capturing good photographs of a moving subject. Due to their small size and mobility, quadcopters are well-suited to act as videographers and are capable of shooting from locations that are unreachable for a human. This paper evaluates the performance of two vantage point selection strategies: 1) a reactive controller that tracks the subject's head pose and 2) combining the reactive system with a POMDP planner that considers the target's movement intentions. Incorporating a POMDP planner into the system results in more stable footage and less quadcopter motion.

Introduction

The "flying selfie bot" has emerged as an important commercial application of quadcopters, due to the voracious consumer demand for high quality photos and videos to share on social media platforms (Schneider 2015). Although most efforts target outdoor and sports photography, quadcopters can be valuable for indoor photographers as well. For instance, Srikanth et al. (2014) demonstrated the utility of quadcopters at providing rim illumination of a moving subject.

To make the best use out of its limited battery life, an autonomous system must solve the same optimization problem faced by a lazy videographer who seeks to produce a steady stream of good footage, with minimal effort, commemorating the event. A practiced human photographer is skilled at finding the best *vantage points* from which to capture the scene. This paper tackles a constrained version of the vantage point selection problem in which the aim is to produce a stream of high-quality frontal photos of a single moving subject over a short time period with minimal quadcopter motion. We evaluate two approaches implemented on a Parrot AR.Drone: 1) a reactive system that tracks the subject's head pose using PD control and 2) coupling the reactive system with a movement policy produced by a POMDP (Partially Observable Markov Decision Process) solver that considers the subject's movement intentions. Head pose is estimated in real-time using a face detector system to calculate a bounding box, along with the facial yaw. The distance between facial landmarks is then used to extract the 3D location of the face relative to the robot's camera. We demonstrate that adding a POMDP planner to the system generates more stable footage. In contrast, the PD controller must constantly respond to the subject's vacillations, resulting in unnecessary motion and lower quality footage.

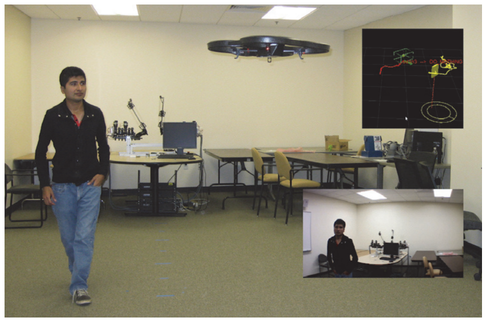


Figure 1: The autonomous quadcopter videographer

Related Work

Autonomous photography can be divided into several subproblems including subject identification, scene composition, and camera positioning. Byers et al. (2003) implemented an autonomous event photographer that followed this procedure to take candid snapshots of conference attendees using an iRobot B2lr mobile robot. Subjects were identified using a color-based skin detection model, and vantage points were selected using an objective function that included a combination of photograph features and reachability considerations. Our evaluation metrics are similar in spirit to this objective function, combining image quality and robot motion. The event photographer was programmed using the same composition rules commonly taught to human photographers. In contrast, Campbell and Pillai (2005) created an autonomous photographer that did not use content-based heuristics for image composition. Instead of relying on color models, a depth clustering process was used to locate subjects based on motion parallax. One advantage of this system is that it could photograph non-human subjects, assuming that they posed for the camera. Since our goal is to produce real-time streaming video footage of a single rapidly-moving subject, our quadcopter videographer does not discard frames or crop images, unlike these systems.

The aim of the related FollowMe system was to do hands-free filming with a quadcopter; for this application, Naseer et al. (2013) demonstrated autonomous human-following using an on-board depth camera. The FollowMe human tracker recovers full body pose from warped depth images, which are also used for gesture recognition. Our proposed system relies on head pose estimation; the human's movement intentions are anticipated using a POMDP, rather than a gesture-based command system.

Our work differs from existing work on tracking humans or vehicles using UAVs (Teulière, Eck, and Marchand 2011) in that our focus is on filming a subject from a specific (frontal) vantage point rather than simply maintaining a subject in the camera's field of view. The problem is more challenging because a small change in the subject's facing can require large quadcopter motions and thus requires more complex motion planning.

Method

Figure 2 shows our system architecture. We use an unmodified, commercially available quadcopter, the Parrot AR.Drone, which comes equipped with two cameras (front-facing and downward-facing), a 3-axis gyroscope and accelerometer, and an ultrasound altimeter. The robot calculates its horizontal speed using vision algorithms (Bristeau et al. 2011) that process data from the downward-facing camera. This information, along with the ultrasound and gyroscope measurements, are sent over WiFi to our PC. In order to command the robot, the quadcopter also receives control messages over WiFi consisting of a desired 3D velocity and the angular yaw velocity, all specified with respect to the robot's current frame of reference. The robot's onboard software translates these commands into rotor speeds to achieve the desired pose.

We employ the ROS framework to handle the communication between the quadcopter and the different parts of our system. At a high level, the *localizer node* tracks the current state of the quadcopter along with the subject's location and relative orientation, while the *controller node* selects vantage points to best capture frontal images of the subject as it moves and generates the appropriate motion commands. Note that a small head turn from the subject can require the quadcopter to sweep in a wide arc in order to photograph the subject from a frontal vantage point.

The *face localizer* sub-module processes the image stream from the quadcopter's front-facing camera to detect faces. If detected, the face's yaw and 3D location are computed and sent to the *publisher*, which transforms the detection from image coordinates to the robot's frame of reference, using the gyroscope data.

The *planner* sub-module calculates a desired robot location based either on a simple reactive plan (i.e., servoing to maintain a frontal vantage point) or a more complex

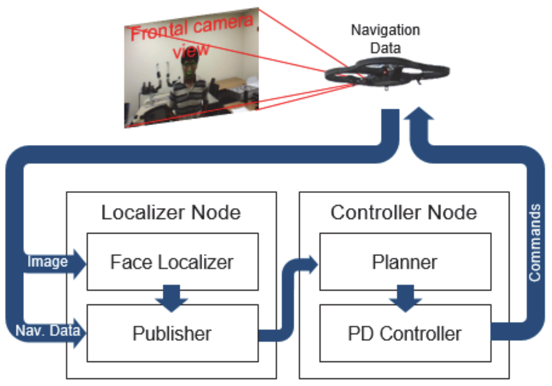


Figure 2: System architecture

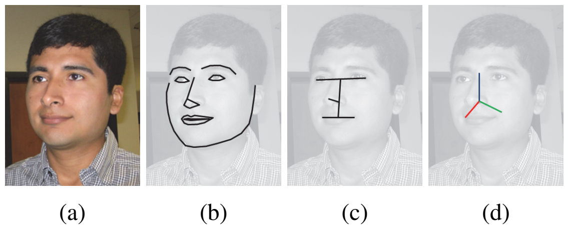


Figure 3: Recovering subject's head orientation: (a) feed from front-facing quadcopter camera (b) landmarks detected by the face alignment process (c) ratios between the five landmarks that determine head orientation. Our method estimates that subject is facing away at a relative yaw of 29°(d).

POMDP policy. The *PD controller* generates the commands to drive the quadcopter to the desired point.

Estimating subject's head orientation and 3D location

To detect the subject's face in the feed obtained from the front-facing camera, we employ a fast detector based on the recent work on max-margin object detector (King 2015), which uses Histograms of Oriented Gradient (Dalal and Triggs 2005) (HOG) features in conjunction with structural SVMs, as made available in the Dlib open-source library (King 2009).

Given the bounding box of a detected face, we use the "millisecond face alignment" method recently proposed by Kazemi & Sullivan (2014) to recover the locations of five key landmarks on the subject's face: the far corners of the eyes and mouth, and the tip of the nose. The method uses regression trees to localize landmarks in real time and we employ the implementation in Dlib (King 2009) that was pretrained on the 300-W faces in-the-wild dataset (Sagonas et al. 2013) without additional training.

With the landmarks located, we calculate the subject face's orientation (relative to the front-facing camera) using the geometric method by Gee & Cipolla (1994), which requires only the ratios of distances between certain facial landmarks to be known. While this method could recover the tilt and yaw of the face, we focus only on the subject's yaw since this dominates the out-of-plane rotation and thus the selection of suitable vantage points. Figure 3 shows an example of the resulting estimate of the pose.

Localizer Node: Publisher Sub-Module

The detected location and pose of the face (obtained with the previous method) are calculated with respect to the camera's frame of reference. However, this needs to be transformed to the robot's frame of reference before computing the control command. Also, since the quadcopter is tilted every time it moves horizontally, the location with respect to the camera may not be accurate. To account for this, we correct the calculated location by the roll and pitch angles that are provided by the quadcopter's gyroscope.

While straightforward in concept, this process is complicated by the fact that the image stream and the navigation data are received at different rates and with different delays. We address this by executing an additional step to match the face location estimate to the most accurate quadcopter location available, inspired by the approach used by Engel et al. (2012). To do this, we maintain a queue of the estimated quadcopter locations over the last second, obtained from the navigation data, and when the face localizer returns an estimate of the face with respect to the camera, we use its timestamp to match the best available estimate of where the robot was when the picture was taken.

We also maintain an estimate of the drone's position with respect to its initial location. This is performed by integrating the (noisy) navigation information, which can result in an inaccurate global estimate of the quadcopter's location. To mitigate the effects of this noise on planning for vantage point selection, the planner operates in the current rather than the global reference frame. The complete state, consisting of location and 2D orientation (yaw) information for the robot and subject (face) is summarized as:

$$(x_r, y_r, z_r, \Psi_r, x_f, y_f, z_f, \Psi_f).$$

Planning Problem - Reactive and POMDP model

Once the pose of the human face is computed, our system must decide on the actions to be executed by the quadcopter in order to capture good video sequences. The subject has the freedom to move anywhere so a straightforward approach is to maintain the quadcopter at a given position with respect to the detected face. This is implemented directly using a reactive model that calculates the goal location in each frame.

Our second approach minimizes the motion of the quadcopter in the case when the subject is shifting slightly around the same central location. This approach was implemented with a Partially Observable Markov Decision Process (POMDP) (Kurniawati, Hsu, and Lee 2008) planner that considers the 'intention' of the human which can be either to change pose or to preserve it. With this information, the controller decides when to change its goal location (similar to the reactive behavior) and when to remain in place.

Reactive Behavior The goal location, which the controller will try to maintain even when the subject is moving, is defined as a point, centered in front of the subject's face.

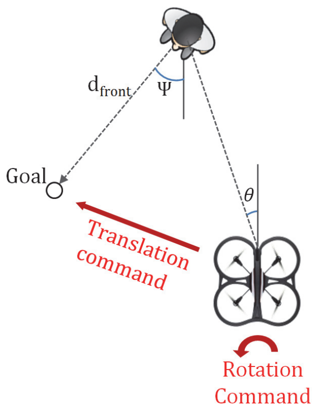


Figure 4: Reactive Behavior: The goal location is calculated as d_{front} meters in front of the person. The height of the goal is d_{height} meters above the location of the eyes.

This is a desirable vantage point from which the quadcopter can obtain frontal photos. This point is determined by two variables:

- d_{front}: the frontal distance between the goal location and the subject face.
- d_{height}: the altitude of the robot with respect to the eyes of the detected face. A value of $d_{\text{height}} = 0$ will maintain the robot at the same level as the eyes of the subject.

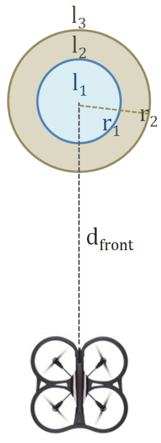
The reactive behavior tries to maintain the robot in the goal location, facing towards the human. In order to do this, a PD controller calculates the specific required commands. For the yaw command, an additional consideration is that the robot should be facing the human at all time (to avoid losing it from its sight). This can be seen in Figure 4, where the command for the robot is to rotate towards the left, although at the final goal location it will be rotated towards the right.

The simplicity of the reactive behavior allows for faster reaction times when the person starts moving, e.g., walking around a room or turning, but the responsiveness comes at a price. Not only is the resulting footage jerkier due to excessive quadcopter motion but it is also possible for the quadcopter to be frequently out of position when the subject turns away and then back, because it responds immediately (with large motions) to small changes in head orientation.

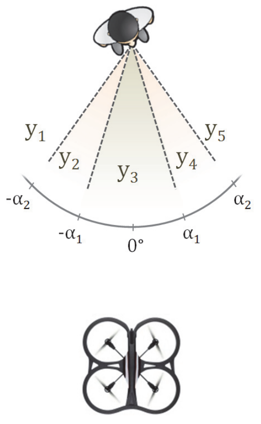
POMDP Planner Behavior The POMDP model attempts to be more judicious in its selection of vantage points by avoiding unnecessary quadcopter motion. A key observation is that a subject may make many small motions near a location punctuated by occasional larger motions between locations. To model this, we predict the *intention* of the subject (either maintaining or changing pose with respect to quadcopter position) and learn a "lazier" controller that executes significant quadcopter motion only when merited.

More specifically, the POMDP quadcopter world has the following partially observable variables:

- loc: This represents the location of the subject, modeled as the distance from the goal, and discretized into three variables: l_1, l_2 and l_3 (see Figure 5a). At each planning step, the goal location is calculated as d_{front} meters in front of



(a) Location model



(b) Yaw model

Figure 5: POMDP quadcopter world: the location is calculated by discretizing the distance from the goal, and the yaw is calculated by discretizing the yaw between -90°and 90°.

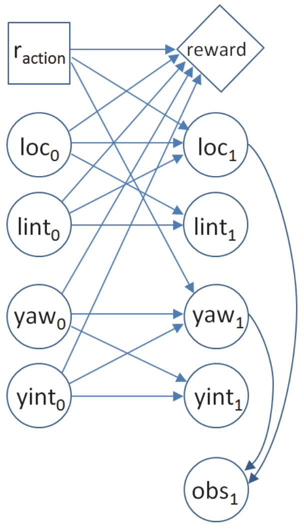


Figure 6: POMDP transition model

the quadcopter. If the distance between this goal (considering only the x and y directions and not the height) and the subject is smaller than r_1, then the value for loc should be l_1. If it is between r_1 and r_2, its value should be l_2. If it is larger than r_2, it should be l_3.

- yaw: The yaw is defined as the angle of the subject's face with respect to the quadcopter's front-facing camera. An angle of zero would mean that the human is facing parallel to the camera (see Figure 5b). These readings are discretized into 5 variables $y_1, \ldots, y_5$ defined by the limiting angles $-\alpha_2$, $-\alpha_1$, α_1 and α_2 ($\alpha_1 < \alpha_2$).
- lint: Represents the "location intention" of the human to move away from its location or not. It can have the values *move* or *keep*.
- yint: Represents the intention of the human to rotate its main focus point (the yaw direction). It can have the values *front*, *rotate_left*, and *rotate_right*.

In each time step, the POMDP planner receives a noisy reading of the location and yaw. This observation is used to update the belief about the current state including the intentions that are not directly observed. For this, we model the transition probabilities as shown in Figure 6. In our model, the subject maintains its location and yaw direction around the safe zones (mostly in l_1 and y_3, but occasionally moving and returning from l_2, y_2 and y_4) when the intentions are *lint* = *keep* and *yint* = *front*. These intentions may change over time causing the human to move away (states l_3, y_1, and y_5). This was pre-defined in the model by setting the probabilities of the subject changing locations higher or lower depending on the intention, and also setting probabilities for the intention itself to change.

At each planning step, the robot may execute two different actions: *do_nothing* or *correct*. The former avoids sending any command to the quadcopter, and the latter calculates a new goal using the same steps as in the reactive behavior and then executes motion commands to achieve it.

We modeled the rewards in the POMDP problem such that the *do_nothing* action is preferred when the subjects' intention is to maintain its pose and its yaw direction. On the other hand, the *correct* action is preferred when the robot location and yaw are in the extremes (l_3, y_1 and y_5) or when the intention has changed, such as when the robot remains in the middle zone too long (l_2, y_2 or y_4).

Finally, in order to connect the planner to the real time system, the POMDP is first solved offline using the SARSOP algorithm (Kurniawati, Hsu, and Lee 2008) to obtain a policy. Then, during the execution, the POMDP controller calculates the best action according to the policy by observing and updating the world state each 0.5 seconds. The resulting action is executed for the following time step. Also, every time the *correct* action was selected, the controller maintains this action for three seconds to complete a full correction rather than a partial movement. The resulting POMDP controller exhibits the expected behavior: short movements of the human do not affect the goal of the quadcopter, so no commands are given, but bigger human movements trigger a corresponding move from the robot. For example, when a human is moving around in the l_1 area, the robot will not move, but if he suddenly moves to the l_3 area, it will move immediately. If the human moves to the l_2 area and stays there, the robot will not move immediately, but after some time, the intention belief updates and it follows the human.

PD Controller

When the goal is calculated (and in the POMDP case, when the controller decides to change it), some commands need to be given to move the quadcopter. These commands are calculated with an underlying PD controller that manages the commands in the x-direction, y-direction, z-direction, and yaw independently. Since the commands need to be given with respect to the frame of reference of the quadcopter, the error (distance to the goal and required yaw rotation) is first transformed to this frame of reference.

In order to get the derivative component of the PD controller, we use the velocities reported from the navigation data (in the x and y direction). The yaw and height components do not need a derivative term, so only the proportional error is used in this case.

Results

In order to test our method, we executed several runs consisting of the controller commanding the quadcopter in a room with a moving subject.[1] The size of the room was 7.5 by 4 meters, with a height of 2.7 meters. We consider three types of runs, characterized by the speed of the subject's movement:

- Scenario Type 1 - Slow Motion: In this case, the subject maintains his position for periods of 15 to 20 seconds, executing short and slow translations (less than 1 meter) and rotations in between.
- Scenario Type 2 - Medium Motion: The subject maintains his position for short periods of time, but now may execute small movements and rotations while doing so. The translations in between are longer (2 to 3 meters) and faster.
- Scenario Type 3 - Fast Motion: The human does not, in general, maintain position. He moves around and rotates his face constantly with higher speed. Occasionally the subject will remain in certain locations for a longer period of time.

The length of each run is exactly five minutes, measured from when the controller switches to an autonomous behavior. The goal of the quadcopter is to be positioned 1.6 meters in front of the human, and at the same height as the eyes. For the POMDP planner case, the values for r_1, r_2, α_1, α_2 are 0.4 meters, 0.6 meters, 15°, and 40°respectively.

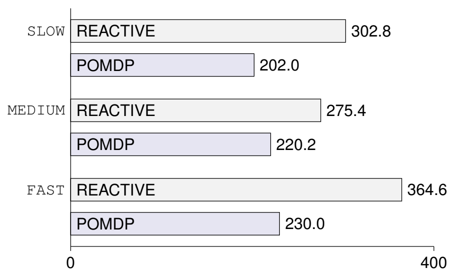
Two categories of metrics were obtained from each run. The first includes command-related metrics that measure the magnitude of the movements the quadcopter executed over time:

- Total Commands x/y/z: the sum of the absolute values of the linear velocity commands in three dimensions given to the quadcopter during a single run.
- Total Commands Yaw: the sum of the absolute values of the rotation commands (yaw) given to the quadcopter during a single run.

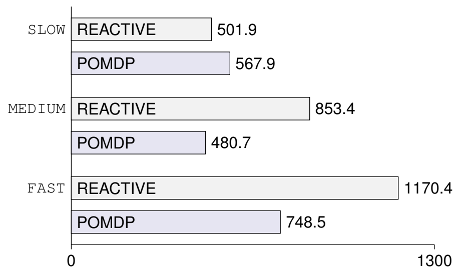
The second includes the face-quality metrics, obtained by analyzing the video recorded by the quadcopter:

- Face Size Error: This metric measures the difference in the size of the face in different frames, with respect to the desired size. The desired size is calculated as the mode of the detected face rectangles in the runs with slow motion. This area represents the size we would expect to see when the robot is precisely at the goal distance. Then, for each frame, we detect the face and calculate the ratio of its area vs. the desired area. If the ratio is smaller than one, we use its inverse. This ratio then represents how different the area is with respect to the desired one. The total error for a run is the average ratio over all frames.

[1]A video of the system is available at: https://youtu.be/s6WJ6SaLZ1k.



(a) Average Total Commands (x/y/z)



(b) Average Total Commands (yaw)

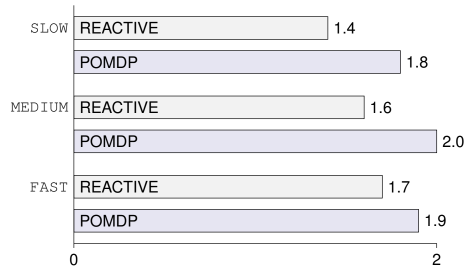
Figure 7: Total commands executed by the controllers under different scenarios. Smaller values indicate that the quadcopter moved less, resulting in more stable movies.

- Face Location Error: This corresponds to the average absolute value of the distance between the face (i.e., the center of the detection rectangle) and the frame center in pixels.
- Face yaw: Average absolute value of the yaw, where zero represents a face that is looking straight at the camera.

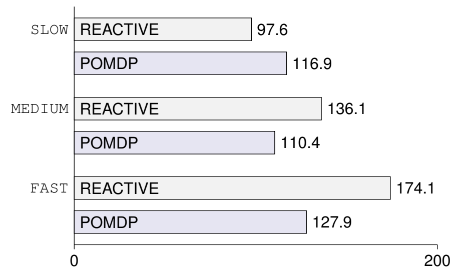
As shown in Figure 7, the POMDP controller generally commands less quadcopter motion in comparison to the reactive behavior. As a result, the videos obtained when using the POMDP planner are more stable and do not shift as much. However, the tradeoff is that the subject may be closer or farther than the desired distance for short periods of time. Figure 8a shows that the face size error is slightly greater with the POMDP controller. This occurs because the POMDP waits longer while determining the subject's intention before executing actions.

Figure 8b shows the error in location of the face, measured in pixels from the center of the image. The POMDP controller exhibits a smaller error for the medium and fast scenarios. This occurs because the subject may be farther than the goal, hence any horizontal movement corresponds to a smaller movement in the center of the detected face. Thus, even though we are allowing the subject to move further before reacting, the face will in average be closer to the center of the frame in comparison to the reactive behavior.

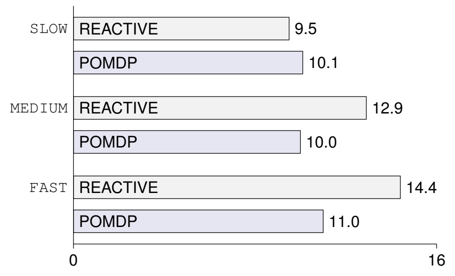
Finally, Figure 8c shows the error in the yaw, measured as the difference in the angle of the observed face vs. a frontal face (looking directly at the quadcopter). The POMDP controller achieves better results in the medium and fast scenarios. By oscillating constantly while trying to maintain itself in front of the subject, the reactive behavior creates situa-



(a) Face Size Error (size difference ratio)



(b) Face Location Error (pixels)



(c) Face Yaw Error (degrees)

Figure 8: Face quality metrics measured as the average difference vs. the ideal size, location (the center of the image), and a frontal looking face.

tions where the subject's head pose is worse. On the other hand, the POMDP filters small changes in the yaw so the average error is smaller.

Conclusion and Future Work

This paper describes an autonomous quadcopter videographer that captures frontal video of the subject. Our solution primarily employs monocular information, which is processed to estimate the subject's facing. We evaluate the performance of two vantage point selection strategies: 1) a PD controller that tracks the subject's head pose and 2) combining the reactive system with a POMDP planner that considers the subject's movement intentions. The POMDP is able to filter short motions and reacts only when the human moves farther or rotates more. As a result, this controller executes less motion, thus obtaining more stable video sequences than the PD controller alone. The ability to capture stable video footage is particularly important for quadcopters used by professional photographers; this is often achieved by adding a gimbaling system, which adds both weight and expense. The POMDP can improve the aesthetic quality of the video in ways that a gimbaling system cannot by anticipating the subjects' rotation and filming from a frontal viewpoint; this differs from commercial quadcopter solutions that simply follow the subject. In future work, we plan to explore more complex image composition policies to shoot group videos. By introducing multi-frame evaluation metrics that consider events rather than static scenes, we can potentially improve the narrative structure of the video in addition to the visual aesthetics.

References

Bristeau, P.-J.; Callou, F.; Vissiere, D.; Petit, N.; et al. 2011. The navigation and control technology inside the AR.Drone micro UAV. In *World Congress of the International Federation of Automatic Control*, volume 18, 1477–1484.

Byers, Z.; Dixon, M.; Goodier, K.; Grimm, C. M.; and Smart, W. D. 2003. An autonomous robot photographer. In *IROS*, 2636–2641.

Campbell, J., and Pillai, P. 2005. Leveraging limited autonomous mobility to frame attractive group photos. In *ICRA*, 3396–3401.

Dalal, N., and Triggs, B. 2005. Histograms of oriented gradients for human detection. In *CVPR*, 886–893.

Engel, J.; Sturm, J.; and Cremers, D. 2012. Camera-based navigation of a low-cost quadrocopter. In *IROS*, 2815–2821.

Gee, A., and Cipolla, R. 1994. Determining the gaze of faces in images. *Image and Vision Computing* 12(10):639–647.

Kazemi, V., and Sullivan, J. 2014. One millisecond face alignment with an ensemble of regression trees. In *CVPR*, 1867–1874.

King, D. E. 2009. Dlib-ml: A machine learning toolkit. *Journal of Machine Learning Research* 10:1755–1758.

King, D. E. 2015. Max-margin object detection. arXiv:1502.00046.

Kurniawati, H.; Hsu, D.; and Lee, W. S. 2008. SARSOP: Efficient point-based POMDP planning by approximating optimally reachable belief spaces. In *Robotics: Science and Systems*.

Naseer, T.; Sturm, J.; and Cremers, D. 2013. FollowMe: Person following and gesture recognition with a quadrocopter. In *IROS*, 624–630.

Sagonas, C.; Tzimiropoulos, G.; Zafeiriou, S.; and Pantic, M. 2013. 300 faces in-the-wild challenge: The first facial landmark localization challenge. In *International Conference on Computer Vision Workshops*, 397–403.

Schneider, D. 2015. Flying selfie bots. *IEEE Spectrum* 52(1):49–51.

Srikanth, M.; Bala, K.; and Durand, F. 2014. Computational rim illumination with aerial robots. In *Proceedings of the Workshop on Computational Aesthetics*, 57–66.

Teulière, C.; Eck, L.; and Marchand, E. 2011. Chasing a moving target from a flying UAV. In *IROS*.