Design and Construction of an Indoor Robotic Blimp for Urban Search and Rescue Tasks

Geoffrey A. Hollinger, Zachary A. Pezzementi, Alexander D. Flurie, and Dr. Bruce A. Maxwell
Swarthmore College
Department of Engineering
Swarthmore, PA 19081

Abstract

This paper describes the construction of a lighter-than-air robotic blimp for use in an urban search and rescue environment. The blimp uses an onboard wireless camera, sonar, and infrared sensors to perform tasks both autonomously and under teleoperated joystick control. During autonomous flight, the blimp is capable of both following lines on the floor and wandering without collision. Additionally, the blimp is equipped with a marker deployment servo to allow the user to mark victims that he or she has identified with the camera. The blimp uses a modular software architecture with separate processes controlling wireless communication, navigation, and vision. Ultimately, this design shows the potential for the use of aerial robots in indoor search and rescue environments.

Problem Description

The blimp described in this paper was developed for use in the Drexel University Aerial Robot Competition. The task for this competition is for the robot to navigate an urban search and rescue obstacle course divided into two parts: autonomous control and teleoperated control. In the autonomous control section, the robot must navigate a simple maze without user input. The task is simplified by the placement of a black line of tape on a white floor denoting a collision-free path. Additionally, the autonomous section contains a vertical ramp and a low-speed fan, meant to simulate wind disturbances.

In the teleoperated section, the robot is controlled by human operators who cannot see the robot, and must therefore rely entirely on the information the robot transmits for information

about the robot and its environment. Several "victims" are scattered about this portion of the gym, and the objective is for the robot to drop markers as close as possible to each victim. The obstacle course described above provides a good sampling of the necessary tasks that a blimp would need to perform in a real urban search and rescue situation.

Introduction

In indoor urban search and rescue environments, it is often necessary for robots to move in the vertical direction to ascend staircases or move between floors. In cluttered indoor environments, such as a collapsed building, airplanes and other low-maneuverability craft cease to be a feasible option. Additionally, most helicopter designs run the danger of destroying themselves during collision with walls or fallen obstacles. The remaining option for aerial search and rescue is a lighter-than-air blimp. Helium is safe and readily available, and current miniaturization allows for sophisticated electronics to be placed on a blimp small enough to be used indoors. Blimp robots also have the advantage of flying at neutral buoyancy without active vertical stabilization, and they are often capable of colliding with obstacles without damaging themselves. Furthermore, wireless cameras and sensing equipment can be easily mounted on blimps to provide important information about unknown environments.

Many aerial robotics labs are not focused at all on blimps. The BEAR lab at the University of California, Berkeley and the UAV lab at the Georgia Institute of Technology work primarily with outdoor rotorcraft. Despite this, indoor airships have had some mention in literature. They have been considered for chassis in work on bio-inspired navigation, and many of the obvious advantages are mentioned [18]. Much of the previous work with lighter-than-air robots, however, has been predominately in outdoor environments. The University of Pennsylvania GRASP Blimp, for instance, uses visual servoing, GPS, and fan encoders to control and localize itself while flying outdoors [11]. This large blimp has a volume of 26 cu. m (the size of a small room) and carries a substantial payload making its use impossible indoors.

When developing indoor blimps, it is often necessary to implement complex behaviors involving decisions based on more information than is immediately available from sensors. To accomplish this, some form of odometry is necessary. Most robot systems use GPS systems or wheel encoders for odometric information [11][18]. Neither of these solutions are options for

indoor aerial robots, since GPS systems do not work in indoor environments and aerial robots have no wheels. It is possible, however, to recover motion information from video, and visual odometry has been used successfully in a variety of robot applications.

Without visual odometry, autonomous operation is also possible by following fixed markers such as lines. The task of line following can be broken up into two parts: extracting the line from an image and sending control signals to move along the line. The choice of the line extraction method constrains the other, since it determines the format of the data that will be sent to the control system.

Line extraction has generally been thought of as edge detection with the refinement of searching for straight edges. Two algorithms have primarily been applied to this problem, the Canny detector [2] and the Hough transform [7]. The Hough transform is a vote-taking scheme which will be described in detail in the Line Following section. The Hough transform has advantages in robustness to noise, ease of implementation, and easy evaluation of the quality of a line detection. Its main disadvantages are that it is computationally expensive, it can only detect lines but not line segments, and the Canny detector better handles the presence of multiple lines in the image. Only the computational expense is a major factor in the line following domain though. For these reasons the Hough transform has become the standard method of line extraction [8], and it has been used in the line following algorithms of several robot systems [3][4]. It was also therefore the method chosen for the system described in this paper.

Hardware Considerations

Blimp Bag and Lifting Agent

When developing the blimp, the primary hardware constraint was the mass it is able to support. It was initially decided to use helium as the lifting agent, but a suggestion was made to use a helium/hydrogen mix that would allow for more lift given the same volume of gas. Helium at STP lifts 1.03 g/L, but balloon bags are over-pressurized, leading to a lift of approximately 1.023 g/L. Hydrogen, on the other hand, lifts 1.12 g/L at STP, or approximately 1.11 g/L when over-pressurized. Hydrogen is inflammable, but care in handling should allow for a small increase in lift potential. The bags used are 52” ellipsoidal mylar balloons. The bags have an

inflated volume of approximately 184 L, which allows for 188.2 g to be lifted for a helium-filled balloon. The bag and fins weight 72.0 g, allowing for 116.2 g of extra lift for components with one bag.

Hardware Overview

Ground-Side

A PC running Linux was used to process sensor data and send control signals to the blimp. A Total Robots Wireless Control Module (WCM) network was used for sensor data communication and blimp-side servo control.[1] Both a remote control that sends the motor control signals and the WCM are connected to serial ports of the PC.

Blimp-Side

The blimp hardware was designed to allow for completion of both the teleoperated and autonomous sections. The undercarriage, blimp motors, and batteries are mounted on the bottom of the blimp with infrared sensors on the sides and a sonar on the front. The WCM and marker deployer are placed behind the undercarriage towards the back of the blimp. The camera is placed at the very front of the blimp. Figure 1 shows a diagram of the blimp configuration.

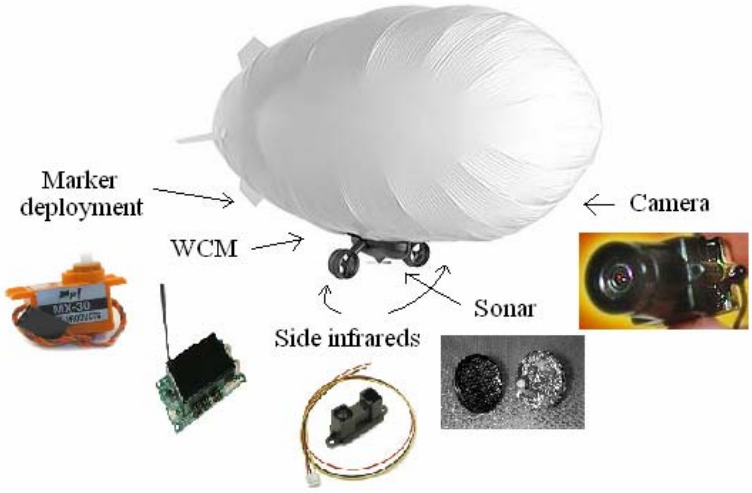

Figure 1: Diagram of blimp configuration

[1] More detailed information about the WCM network may be found in its own section.

Hardware for use on the blimp was chosen based upon its utility in simple line- and space-following configurations and its weight. The sonar has a wide window of detection, ranging from a few centimeters to 4 meters and is lightweight. The IRs are also lightweight and long range, reaching as far as 2 meters. The eyecam was an extremely lightweight blimp-side package and offered the best quality in range and picture of the possible cameras within the budget of the project. A small picoservo was used for the deployment of the rescue markers in the teleoperated section of the course.

The picoservo, a plastic cross-shaped loading carriage, and a plastic tube make up the teleoperated section's marker deployment system. This system relies on the servo's ability to simultaneously load and launch a marker in one motion. This is achieved by creating markers that are the exact size of the loading carriage attached to the servo such that only one marker is loaded at a time. The motion is sufficiently quick that extra markers do not deploy while one is being launched, and the carriage is walled sufficiently to avoid the unwanted dropping of badly-positioned markers.

Hardware Configuration

The barebones configuration for manually-operated flight requires only the undercarriage and sufficient ballast to bring the craft to neutral buoyancy. This allows manual control of a balloon using the remote, and this configuration is implemented during the manual control section of the competition.

The configuration for simple teleoperated control requires the camera, the 9V battery for powering the camera, the undercarriage, and any extra ballast needed. This allows teleoperated control of a balloon using the nav module and the computer joystick interface. The addition of the WCM, the 6V battery, and the picoservo marker deployment system allows the teleoperated, search-and-rescue section of the course to be completed.

The configuration for autonomous line-, space-, or wall-following requires the camera, the sonar, the IR sensors, the 9V battery, the undercarriage, the WCM, and the 6V battery. The only piece of equipment not vital to autonomous functionality is the marker deployment system.

Wireless Communication Module

To transmit sensor data and deployer positions between the blimp and the host computer, a wireless communication module (WCM) was used. In autonomous mode,[2] the mode used in this paper, the registers of the blimp WCM mimic those of the WCM connected to the server computer through the RS232 port. The WCM network transmits information wirelessly at 2.4 GHz and allows for up to four analog sensors and eight digital I/O devices (including servo motors). On the server side, the host computer communicates with the WCM asynchronously using a serial protocol. The host computer can both write to and read from the WCM using this protocol. Additionally, the WCM weighs less than 40 grams, making it ideal for use on the blimp. Figure 1 gives a picture of the WCM and a diagram of its operation.

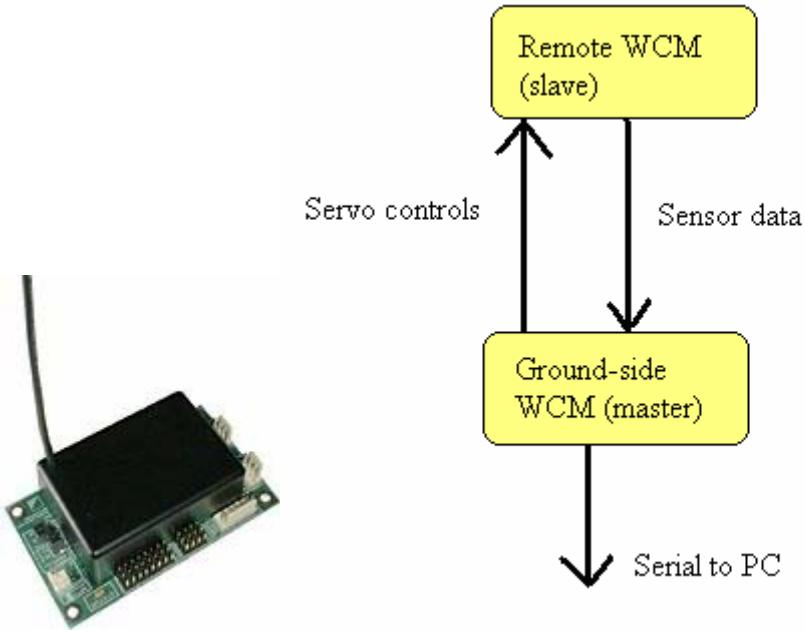

Figure 1: Picture of WCM (left) and diagram of WCM operation (right)

In the final blimp configuration, two infrared sensors and one sonar were connected to the WCM's analog input ports. The servo motor for marker deployment was connected to one of the I/O ports. Using the WCM software module, described in the next section, the host computer calls for the reads and writes necessary to retrieve sensor data and deploy markers.

[2] Other modes are available for the WCM. For more information, examine the datasheet at www.totalrobots.com.

Host Computer Software Architecture

In the design of the blimp's software architecture, it was necessary to separate the visual processing, navigation control, and wireless communication because a simple program could not perform all of these tasks simultaneously without unnecessary delay. To optimize the operating frequency of each task, a modular software architecture was implemented. The navigation module handles communication with the blimp's motors and makes control decisions during autonomous operation. The wireless communication module reads sensor data and writes servo positions to the wireless device on the blimp. Finally, the vision module, handles image processing and line-following. Figure 2 gives a diagram of the blimp module structure.

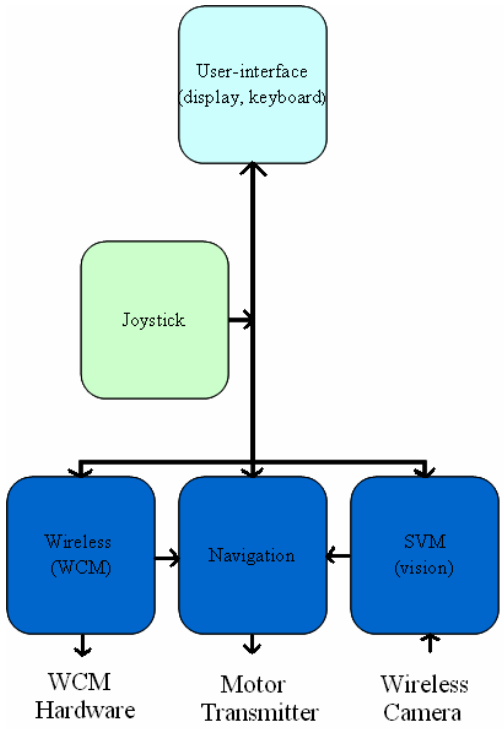

Figure 2: GCM module structure of blimp

Each of the processes described above were set to communicate using the IPC network protocol [15]. The IPC protocol has been successfully applied to multi-robot systems and other robot communication tasks [12]. The use of IPC communication for the blimp software was facilitated by the use of the General Communications Module (GCM) developed at Swarthmore College [6]. GCM allows separate processes to asynchronously share information about the state

of the robot without delaying their operation. On the blimp, each module is allowed to run at a fixed frequency without unnecessarily waiting for updated information from the other modules. If the wireless communications module stops sending information, for instance, the navigation and vision modules are able to continue operation.

Teleoperated Joystick Interface

To perform the tasks necessary for the teleoperated portion of the competition, a joystick interface was implemented using the SDL (Simple DirectMedia Layer) library. SDL allows for low-level access to the joystick by handling events such as button press and axis motion. SDL functions give the blimp software direct access to the state of joystick buttons and the positions of the axes. A Logitech Wingman joystick was used to operate the blimp, and buttons on the joystick were specified to drop markers and to move in the vertical directions. Additionally, a linear function of the positions of the joystick axes was used to determine the rotational and translational velocities of the blimp. Using GCM, the joystick module is able to easily communicate with the wireless communication module and the navigation module. This implementation provides an intuitive interface for the user to operate the blimp while viewing the wireless video. Figure 3 shows a diagram of the joystick mappings.

Figure 3: Joystick interface for teleoperated blimp control

Visual Line-Following

In order to navigate the autonomous portion of the course, a line following algorithm was implemented. The line following task can be divided into two parts, that of detecting the line and that of controlling the robot to follow the line. The line extraction was accomplished using a Hough transform as described below, and the controller went through two incarnations.

Figure 4. Input grayscale image (left), + and - gradient images (middle and right)

The line extractor takes as input an image from the camera mounted on the front of the blimp. First, this image was converted to grayscale. Sobel-x gradient filters were then applied to the image two create two edge images, one of large positive gradients and one of large negative gradients (Figure 4). Hough transforms were applied to each of these edge images, defining a line by angle of orientation and perpendicular distance from the center of the image. A Hough transform is a pattern-matching algorithm that works through vote-taking. The pattern searched for was a line in each of the edge images. The range of angles 0 to 2π was discretized as well as the range of possible perpendicular distances within an image. These two ranges defined the x- and y-dimensions of the Hough accumulator, which allows the parametrization of a line by its angle with respect to the standard coordinate system and its perpendicular distance to the center of the image. Each "on" pixel in the edge image then voted for every line which could possibly pass through it, incrementing the corresponding slot in the Hough accumulator, as shown in Figure 5. The strongest lines in each edge image are then found as the maximal values in the Hough accumulators. When trying to find multiple lines within the same image, one must take care not to find the same line multiple times. After making each selection, the area of the accumulator immediately around the selection is therefore suppressed.

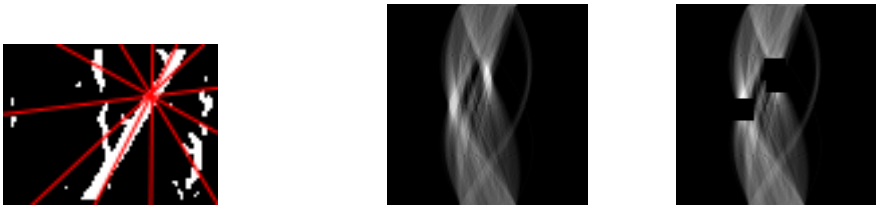

Figure 5. Votes made by one sample "on" pixel (left), a sample Hough accumulator (middle), and the same accumulator after two rounds of line selection and local suppression. For both Hough accumulators, the y-dimension corresponds to angle and x to distance from center.

The best t_1 lines were then found in the transform of each edge image by finding the largest value in the Hough accumulator then zeroing all values in a local neighborhood around that point. This set of t_1x t_1 lines was searched for all pairs who met two criteria. The lines must be nearly parallel, $fabs(lines1_i.angle - lines2_j.angle) < t_2$, where lines1 and lines2 are the sets of the best line matches for the two edge images. The negative gradient must occur to the left of the positive gradient as well, $lines1_i.dist > lines2_j.dist$, in order to detect the left and right sides of a dark line on a light background. Finally, an average line is computed from all pairs meeting these criteria, weighting each line's components (angle and distance) by its pixel support from the Hough accumulator. These average angle and distance measurements are sent to our navigation module along with the sum of the pixel supports of the lines that were averaged. If this support is above a thresholded value, t_3, a line is said to be detected. Otherwise the track is considered to have been lost. The width of the detected average line is also computed as the weighted average of the difference between the distance values of the constituent lines.

The first line following control algorithm was a finite state machine which simply tried to keep the line in the center of the image with an orientation close to vertical. If the line was to the right of the center of the image and also has a positive angle with respect to the vertical, then the blimp would rotate in place to the right. If the line was instead left of center but still has a positive angle with respect to the vertical, the blimp would move forward while veering right. Similarly, if the line was left of center and had negative angle, the blimp rotated left, and in the remaining case it moved forward while veering left. Whenever the line is lost, the blimp would turn in place in the hopes of finding the line again. This controller turned out to be woefully inadequate. Its major disadvantages were a lack of precision and of any way to deal with the

blimp's momentum as well as poor control of the vertical positioning of the blimp with respect to the line.

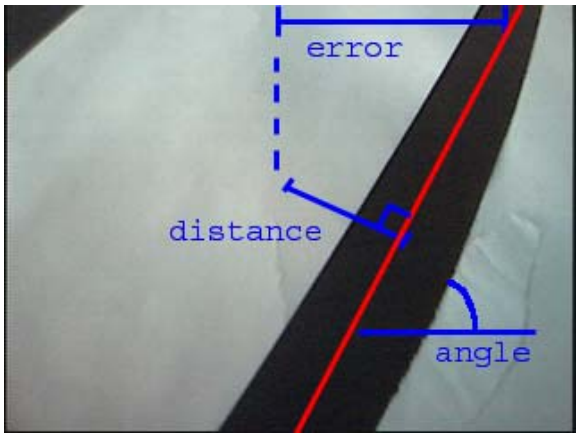

Figure 6. Conversion of line angle and distance to an error metric for PD control. Dist represents the line's perpendicular distance from the center of the image and error is the input to the controller.

All of the issues with the first controller were addressed in the design of the second controller. The detected line was intersected with the top of the image to find the x-distance of this intersection from the top center of the image, giving a measurement of error in heading, as shown in Figure 6. This error was then used as the input for a PD controller for rotation of the blimp. The derivative term was included to damp the inertia of the blimp since the previous controller had a great deal of trouble with the blimp continuing to rotate long after any rotation commands had been given. The forward control was implemented to go more slowly when the rotation controller gave a large output and more quickly when not rotating. An up-down controller was added, implemented as a P controller which was given as input the difference between the width of the detected line and a threshold value (the width corresponding to the desired height). Additionally, when the line is lost, the blimp was programmed to fly up in the hopes of being able to find the line again with a larger field of view while turning back in the direction the line was last seen.

Competition Performance

At the Drexel Aerial Robot Competition, the blimp's performance in the autonomous section was exceptionally good. The blimp was able to traverse the entire autonomous section of the course up to the gust stabilization fan without using any of the two allowed restarts. Once the blimp reached the fan, it was unsurprisingly blown completely off-course, since we had no active gust stabilization and the fan used in that portion of the course was much stronger than any stabilization system that could have conceivably been carried by the blimp anyway. The Drexel blimp was able to reach the fan portion of the course on their best run as well but not without restarts. Their blimp was equally incapable of dealing with the fan.

During the teleoperated portion of the competition, the blimp had several problems that inhibited its performance. After successfully locating and marking one victim, the blimp hit a cold area of the gym and was no longer able to maintain vertical stability. This was due to the low power of the vertical motor on the blimp. A redesign of the motors to provide more vertical force would solve this problem. The joystick interface and deployment mechanism, however, performed well.

Conclusions

The success of blimp described in this paper in the autonomous section of the Drexel Aerial Robot Competition shows the potential of lighter-than-air robots for use in urban search and rescue environments. The problems in the teleoperated section were due to underpowered motors, and they could be solved through a simple motor redesign. Overall, it was determined that blimps are excellent robotics platforms, and they have many potentially useful applications.

Future Research

For future research, a full redesign of the motors is suggested. The blimp motors, particularly the vertical motor, were severely underpowered in the teleoperated environment. Larger motors and a custom battery would solve the vertical stabilization problems. A custom transmitter and larger blimp bag would also improve the blimp's performance in the teleoperated

section. Furthermore, a sliding autonomy component should be added to the teleoperated interface. This would allow for the blimp to act autonomously for some tasks (like searching for victims) and then prompt the user to give more specific movement commands and deploy the marker. Sliding autonomy has been successfully used by [1] for robotic structure construction, and its application to aerial robots would potentially allow the user to control several blimps at once.

Additionally, further work is suggested on the autonomous space-following algorithm and visual odometry. With more accurate visual odometry, the blimp could act autonomously without a line. This would greatly improve the potential applications and allow for operation in environments that have not been specifically setup for the task.

Acknowledgment

Thanks to Ed Jaoudi for electrical work and to Grant Smith for machine shop consultation. Further thanks to Nicholas Ward and Fritz Heckel for assistance with the GCM interface.

References

[1] J. Brookshire, S. Singh, and R. Simmons. “Preliminary Results in Sliding Autonomy for Coordinated Teams.” Proceedings of The 2004 Spring Symposium Series, March, 2004.

[2] Canny, J., “A Computational Approach to Edge Detection,” IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol 8, No. 6, Nov 1986.

[3] Chen, Haoqian and Elena Glassman, Chengjou Liao, Yantian Martin, Lisa Shank, Jon Stahlman. "AIBO Motion and Vision Algorithms," 2005. http://www-2.cs.cmu.edu/~tekkotsu/media/pgss_2003_paper.doc.

[4] Chen, K. H. and W. H. Tsai, “Autonomous land vehicle guidance by line and road following using clustering, Hough transform, and model matching techniques,” Proceedings of 1994 International Computer Symposium, Hsinchu, Taiwan, Republic of China, pp. 89-94.

[5] Hartley, R. and A. Zisserman. Multiple View Geometry in Computer Vision. 2nd ed. Cambridge University Press, 2003.

[6] Heckel, Frederick. "General Communication Module (GCM)", 2005. http://robotics.swarthmore.edu/gcm/about.shtml.
[7] Hough, P.V.C., “Machine Analysis of Bubble Chamber Pictures,” International Conference on High Energy Accelerators and Instrumentation, CERN, 1959.
[8] Illingworth, J. and J. Kittler, “A survey of the hough transform,” Computer Vision, Graphics, and Image Processing, 44, 1988.
[9] Jongwoo Kim, James P. Ostrowski, “Motion Planning of Aerial Robot using Rapidly-exploring Random Trees with Dynamic Constraints,” IEEE Int. Conf. on Robotics and Automation, Sep, 2003.
[10] Jongwoo Kim, Jim Keller, Vijay Kumar, “Design and Verification of Controllers for Airships,” IEEE IROS, Oct, 2003.
[11] Kumar, Vijay, “University of Pennsylvania Blimp Specifications” http://www.cis.upenn.edu/marsteams/GRASP%20Lab%20Blimp%20specifications.pdf
[12] Melchior, Nik and William D. Smart. “A Framework for Robust Mobile Robot Systems.” In Proceedings of SPIE: Mobile Robots XVII, October 2004.
[13] D. Nistér, O. Naroditsky and J. Bergen, Visual odometry, Proc. IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR 2004), Volume 1, pages 652-659, 2004.
[14] Owens, Robyn. “Epipolar Geometry” http://homepages.inf.ed.ac.uk/rbf/CVonline/LOCAL_COPIES/OWENS/LECT10/node3.html
[15] Simmons, Reid, Carnegie Mellon University. "Inter Process Communication (IPC)", 2005. http://www-2.cs.cmu.edu/afs/cs/project/TCA/www/ipc/ipc.html.
[16] Taylor, C.J. and James P. Ostrowski, “Robust Visual Servoing based on Relative Orientation,” IEEE ICRA 1999.
[17] Zhang, Hong and Jame P Ostrowski, “Visual Servoing with Dynamics: Control of an Unmanned Blimp,” IEEE ICRA 1999.
[18] Zufferey, Jean-Christophe. *Bio-Inspired Vision-Based Flying Robots*. Ecole Polytecnique Doctoral Thesis, 2005.

Appendix A: Parts List

Two wireless communication module for two-way communication of ir, sonar, and servo positions between the blimp and host computer (WS-WCM, $100 each):

http://www.totalrobots.com/access_files/wcm.htm#wcm

Two long range IR sensors used on the sides of the blimp (R144-GP2Y0A02YK, $15 each):

http://www.acroname.com/robotics/parts/R144-GP2Y0A02YK.html

Analog sonar with onboard driving mechanism used on front of blimp (MUSTA 8V-12V, $80):

http://www.mrrobot.com/

Blimp bags (52'' blimp balloon, $14), motors (tri-turbofan blimp, $70), and telecommander control software (telecommander RS-32, $25):

http://www.plantraco.com/price_list.html

Draganfly 2.4 GHz wireless eyecam ($250):

http://www.rctoys.com/eyecam.php

Batteries – Zukz rechargeable CR123A 3V ($20 with charger), Ultralife lithium 9V ($10 each), and Energizer lithium L544 6V ($8 each):

http://www.batteryprice.com/

Pico servo (R54-MX-30) used for marker deployment ($25):

http://www.acroname.com/robotics/parts/R54-MX-30.html

Logitech Wingman Extreme joystick ($25):

http://www.amazon.com

Appendix B: Visual Odometry

Visual odometry was implemented according to the methods described in Hartley-Zisserman [5]. Essentially, one starts with two views of the same scene, such as the image pair in Figure 8, and one tries to recover the translation vector T and rotation matrix R shown in Figure 7.

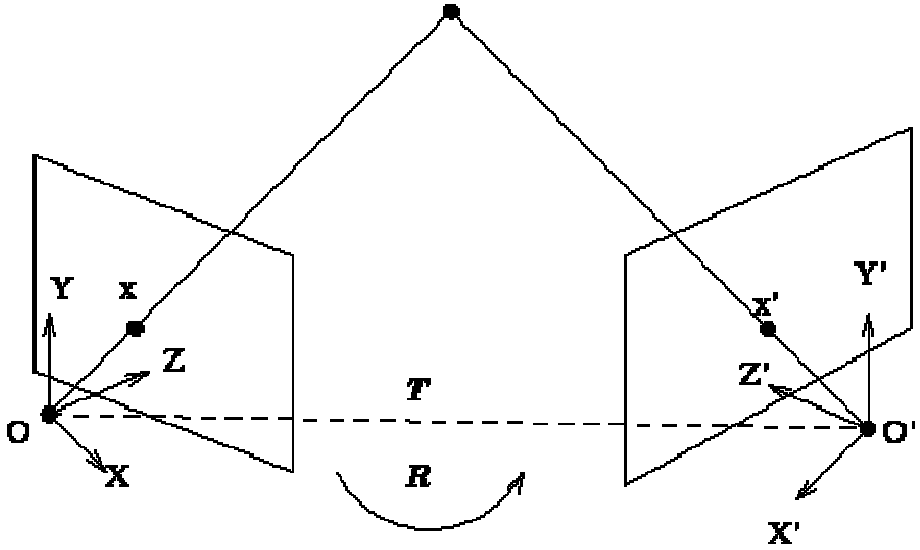

Figure 7. Geometry of a two-view camera configuration. O and O' represent the 3D camera positions and XYZ/X'Y'Z their respective coordinate frames. x and x' are the projected image locations of the unabeled 3D world point at the top. T and R represent the translation vector and rotation matrix respectively associated with moving from the left camera frame to the right camera frame. [13]

In order to recover three-dimensional information from two-dimensional images, one must first compute the fundamental matrix F. For a set of point correspondences between two images $x_i \Leftrightarrow x_i'$, where x_i and x_i' are homogenous image coordinates of corresponding points in each of two images, $x_i'^T F x_i = 0$. Through this formulation, F is said to encode the epipolar constraint: the world point whose image maps to a pixel in one image can only appear along a ray in the other image. The end point of that ray is defined by the intersection of the baseline, the line connecting the two camera centers, with the image planes, as shown in Figure 8. Interest points to use for these correspondences can be obtained through the use of a Harris corner detector, as shown below.

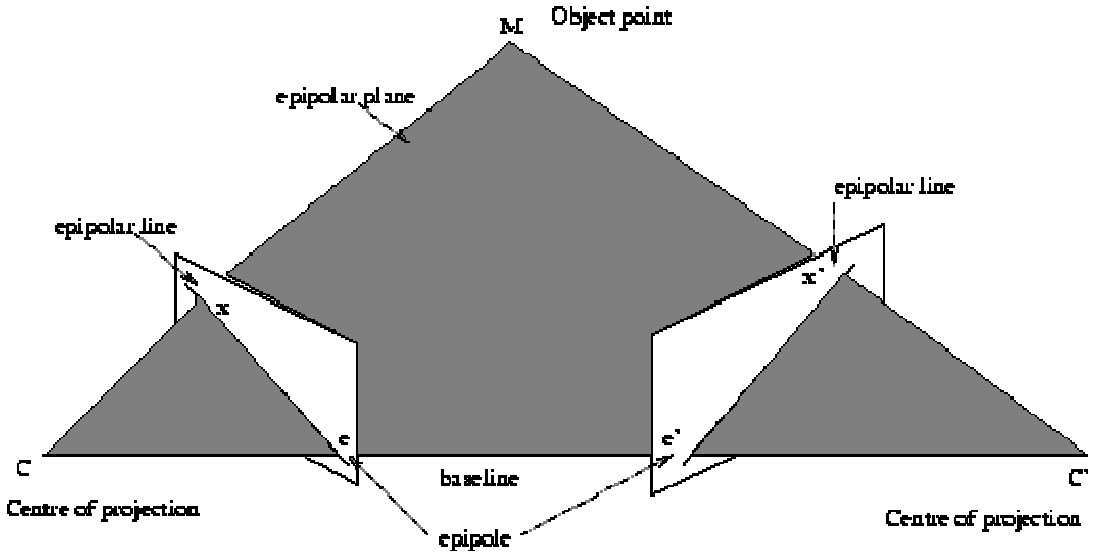

Figure 8. Epipolar geometry. [13]

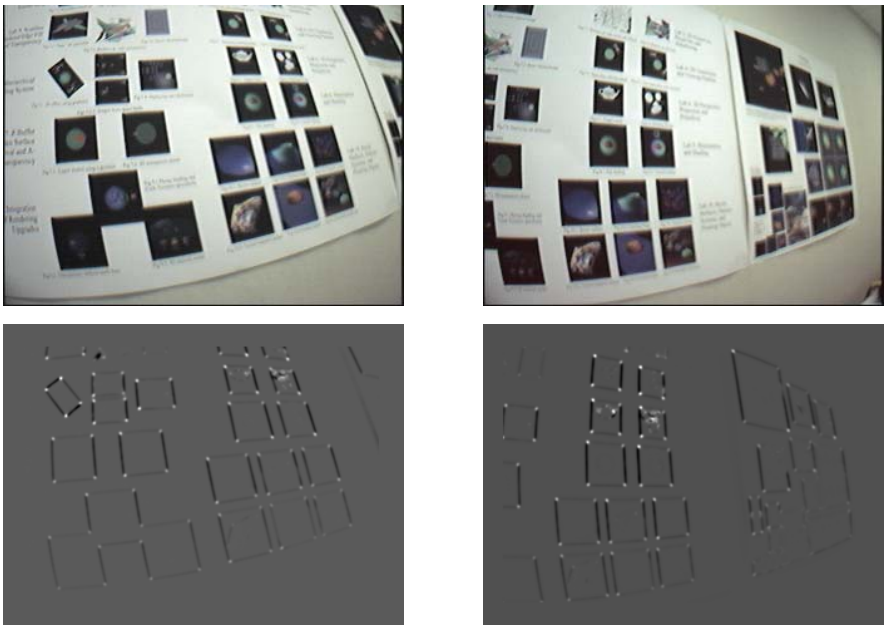

Figure 9. Two views of a poster taken with the same camera in different positions (top) and the output of the corner detector for those images (bottom).

A first estimate of F is found by constructing A as

$$A = \begin{bmatrix} x'_1x_1 & x'_1y_1 & x'_1 & y'_1x_1 & y'_1y_1 & y'_1 & x_1 & y_1 & 1 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ x'_nx_n & x'_ny_n & x'_n & y'_nx_n & y'_ny_n & y'_n & x_n & y_n & 1 \end{bmatrix}$$

Eight point correspondences are necessary to constrain the eight free parameters of the fundamental matrix (3x3 defined up to a scale). A first estimate of F is found as the singular vector corresponding to the smallest singular value of A, as in Algorithm 11.1 of Hartley-Zisserman. F is itself decomposed and its smallest singular value set to zero to enforce the constraint that $\det F = 0$.

Next, a set of point correspondences (possibly the same as those used above) is corrected to minimize the geometric error when computing the epipolar constraint, as described in algorithm 12.1 of Hartley-Zisserman. Translation matrices are computed to bring each point to the origin, and rotation matrices to align the coordinate axes with the left and right epipoles. A formulation of F as

$$F = \begin{pmatrix} ff'd & -f'c & -f'd \\ -fb & a & b \\ -fd & c & d \end{pmatrix}$$

allows for the construction of the sixth degree polynomial

$$g(t) = t\left((at+b)^2 + f'^2(ct+d)^2\right)^2 - (ad-bc)(1+f^2t^2)^2(at+b)(ct+d) = 0\,.$$

The cost function, $s(t) = \dfrac{t^2}{\left(1+f^2t^2\right)^2} - \dfrac{2(ad-bc)(at+b)(ct+d)}{\left(\left(at+b\right)^2 + f'^2\left(ct+d\right)^2\right)^2}$ is then evaluated at each root of g and at $t = \infty$ and the minimum value, t_{min}, is selected. Two lines are constructed as

$$l = \left(tf, 1, -t\right) \text{ and } l' = \left(-f'(ct+d), at+b, ct+d\right)$$

The closest point on a line (λ, μ, v) to the origin is given by $\left(-\lambda v, -\mu v, \lambda^2 + \mu^2\right)$, which is evaluated for l and l'. These points are finally transformed back to the original coordinate system through multiplication by the inverse transformation and rotation matrices to get the corrected point correspondences.

The three dimensional world space coordinates X_i can then be recovered by finding estimates of P and P', the camera projection matrices into the first and second image respectively. B is constructed from p^{iT} and p'^{iT}, columns of P and P', as

$$B = \begin{bmatrix} xp^{3T} - p^{1T} \\ yp^{3T} - p^{2T} \\ x'p'^{3T} - p'^{1T} \\ y'p'^{3T} - p'^{2T} \end{bmatrix}$$

and the solution is found as the singular vector corresponding to the smallest singular vector of B.

The Maximum Likelihood estimate of the fundamental matrix can then be computed by the Gold Standard method, Algorithm 11.3 of Hartley-Zisserman, using the initial estimates calculated above. F is refined by Levenberg-Marquardt minimization of the cost function

$$\sum_i d\left(x_i, \hat{x}_i\right)^2 + d\left(x_i', \hat{x}_i'\right)^2$$

where $\hat{x}$ denotes the projections of the altered 3D points $\hat{X}$. Minimization is done on 36 variables, three for each of the 3D points and 12 for the camera matrix P', initialized to $P' = \left[\left[e'\right]_x \hat{F} \mid e'\right]$, where e' is the left epipole and $[]_x$ denotes the skew-symmetric matrix (P is set to $\left[I \mid 0\right]$). Computation of the Jacobian matrix of the cost function is carried out numerically by incrementing each variable x_i by δ, where $\delta = \max\left(\left|10^{-4} * x_i\right|, 10^{-6}\right)$. The selection of the point correspondences will be extended to a RANSAC procedure to improve our estimate of F. Translation can finally be recovered as the fourth column of P' and rotation from the first three columns.

The current implementation uses pre-existing (and therefore working) code for interest point extraction and correspondence of those points between images. The rest of the code is not however verified to give correct output. The implementations of individual equations given above have been checked to ensure that they impose the mathematical constraints they intend to, but there was no obvious way to do validations like checking output for estimates of the fundamental matrix against physical ground truth. It is possible to obtain validation data consisting of a pair of images and accurately-measured camera positions from which each image was taken. This data only allows one to check the performance of the complete system though, not the individual components. Debugging of the system could be greatly facilitated if the results of intermediate calculations of a similar working system were available.