17th Mediterranean Conference on Control & Automation
Makedonia Palace, Thessaloniki, Greece
June 24 - 26, 2009

Vision Aided Navigation for Unmanned Helicopters

Diomidis Katzourakis, Nikos I. Vitzilaios, and Nikos C. Tsourveloudis

Abstract— **The development of a vision system to aid the autonomous navigation of an unmanned helicopter, primarily based on inertial sensors and GPS data, is presented. An unmanned helicopter has been equipped with appropriate sensors and a vision system fitted on a custom pan-tilt mechanism. Robust software, based on the Open Computer Vision Library (OpenCV), has been developed for handling images and video from a camera. Our implementation involves real time object recognition, histogram matching for real time video streaming, pattern matching and object tracking. Software implemented in C++ interacts with Matlab in order to aid the autonomous navigation of the helicopter.**

I. Introduction

THE Scale Invariant Feature Transform (SIFT) is an algorithm to detect feature points in images, commonly called keys or keypoints in the SIFT framework. The primary goal of the SIFT algorithm is object recognition. Harnessing the Gaussian difference of the images in different scales the SIFT algorithm can ensure scale invariance; by assigning one or more orientations to each keypoint location based on local image gradient directions can provide rotation invariance and through other techniques can provide features that are translation invariant, which could also be partially invariant to illumination changes and affine or 3D projection [1]. The application of SIFT algorithm on an image has as outcome a keypoint descriptor for representing points of interests in order to be matched against a database. The algorithm is considered to be very robust and keypoints have been invariant across a substantial range of affine distortion, addition of noise, and change in illumination [1].

SIFT has been widely accepted in the community of machine vision mainly for its repeatability, distinctiveness, and robustness. SIFT has also been widely utilized by Unmanned Aerial (UAV) and Ground (UGV) vehicles research teams [2].

This work was supported in part from the 03ED365 research project, implemented within the framework of the "Reinforcement Programme of Human Research Manpower" (PENED) and co-financed by National and Community Funds (75% from E.U.-European Social Fund and 25% from the Greek Ministry of Development-General Secretariat of Research and Technology).

Diomidis Katzourakis was with the Technical University of Crete, Intelligent Systems & Robotics Laboratory, 73100, Chania, Greece. He is now with TU Delft, Biomechanical Engineering, 2628 CD, The Netherlands (e-mail: d.katzourakis@tudelft.nl).

Nikos I. Vitzilaios and Nikos C. Tsourveloudis are with the Technical University of Crete, Intelligent Systems & Robotics Laboratory, 73100, Chania, Greece (e-mail: {vitzilaios, nikost}@dpem.tuc.gr).

Lin et al. [4] have used the SIFT algorithm for the registration of consecutive images taken from a UAV into a mosaic. They tried many kinds of features but concluded that the SIFT features, due to the properties mentioned earlier, would be the most appropriate solution for the concept. The SIFT algorithm, has also been used by Forrstner et al. [5] for online geocoding of large scale images using a video camera on a UAV. Abdallah et al. [8] used SIFT in a vision-inertial SLAM system for UGV, using a camera for the detection of landmarks on the ground and an IMU for dead-reckoning. They estimate the robot motion by dead-reckoning information in an attempt to predict future positions of the SIFT features during navigation. SIFT feature tracking has also been used by Templeton et al. [6] for the evaluation of their vision-based terrain mapping and analysis system for an unmanned helicopter.

Another scheme for feature extractions is the Speeded Up Robust Features (SURF), a high performance algorithm which provides interest point detector and descriptor invariant in scale and rotation [9]. SURF is considered to be as thriving as SIFT and yet can be computed and compared much faster [9]. The software for the SURF algorithm is provided in [10].

A similar provision to ours for UAV machine vision using SIFT, has been built in [2] and [3], using the same software we did, as a basis for developing the vision system [11]. Their research aims on techniques for autonomous flight using mainly visual input source, since as they assert, the GPS is responsible for guiding the UAV to the vicinity of the desired target. From that point, vision is the main source for piloting. Because their implementation resembles to ours we will collate it along with ours.

II. Hardware Implementation

An unmanned helicopter has been developed for experimental use and testing on autonomous navigation techniques (Fig. 1). The unmanned helicopter is equipped with all appropriate sensors for autonomous navigation (Inertial Measurement Unit, GPS receiver, RC servo driver\controller, CPU unit, Solid State Disk) according to our previous experience on the field [12]. A custom made pan-tilt vision system is mounted on the helicopter.

The pan-tilt vision system consists of a webcam and two mini servos, controlled through a custom built servo controller based upon an ATmega8 AVR microcontroller from ATMEL (Fig. 2). The system interacts with the computer with two USB wires; the first for the webcam and the second for the servo controller. The microcontroller communicates with apc, through a handy USB to serial chip from FTDI [1] that provides the necessary voltage supply for the system.

Fig. 1. Unmanned helicopter equipped with vision system.

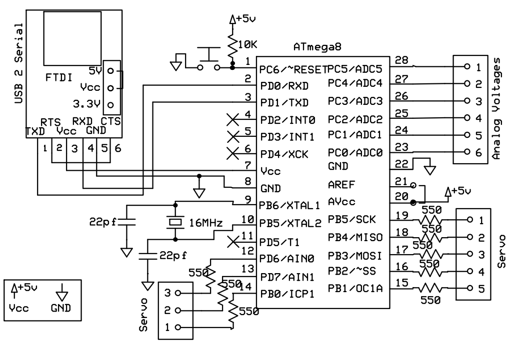

Fig. 2. Servo controller for the camera pan/tilt movement.

The servo controller has been built for reading (and writing) raw data (bytes or characters) to the computer. The writing has to do with the fact that the controller also serves the role of an analog to digital converter (Fig. 2). We used raw data against ASCII for higher bandwidth. The serial protocol that has been built is shown at Fig. 3. It is special character (ASCII) driven. For our case the special character was the raw value of '$'(which is the $36_{decimal}$). The characters following a special character are: the start of frame 's', data indicating zero value which is the character '0' and the character notifying that the original data has the value of the special character ($36_{decimal}$) which is the '^'. That is:

- to start a new data frame→ '$','s'
- send a zero value for data→ '$', '0'
- finally in case we want to send the raw decimal value of the special character ('$'=36)→ '$', '^'.

The controller accepts byte data as mentioned above. Every byte is a decimal number which multiplied by the time quantum of 16usec, representing the high voltage for the PWM signal. For example, if a byte with the decimal number 93 is received, it will produce a PWM signal 93x16usec= 1.5msec for the high pulse and 18.5msec for the low pulse.

III. Software Implementation

Our implementation is based upon the Open Computer Vision Library (OpenCV) [14] for real time computer vision. OpenCV is an open source computer vision library in C/C++, independent from the operating system and hardware, is optimized and is suitable for real time applications [15]. The software is written in C++. The concept of the project is to aid a UAV's guidance based on inertial sensors and GPS with machine vision. More specifically, we have implemented real time object recognition, using the SIFT algorithm [1] and histogram matching for the captured frame. The Scale Invariant Feature Transform (SIFT) is an algorithm to detect feature points in images. It uses a class of scale, rotation and translation invariant features, which could also be partially invariant to illumination changes and affine or 3D projection [1]. In our case, these features are used for matching between a still image (acting as pattern) and a captured video frame, performing object detection and computing their between geometrical transformation. Our program is based upon an implementation of the SIFT algorithm [11]. The flow chart of the program is illustrated at Fig. 4.

If the captured image surpasses a user defined threshold of the minimum "SIFT features", the program tries to match the SIFT keypoints from the pattern we are searching with keypoints from the capture frame. This identification of the most similar keys is considered to be high in complexity [16], thus a modified version of the k-d tree algorithm is used, called the best-bin-first search method proposed by Beis & Lowe [17], laconic in computation resources though with high effectuality the nearest neighbors.

When the SIFT keys are matched, the program finds the pattern's approximate two nearest neighbors in the k-d tree. If they satisfy a user-defined threshold on squared ratio of distances, the RANSAC [15] algorithm is executed for the calculation of a best-fit image transform between the corresponding SIFT keys for the pattern and the captured frame.

The RANSAC algorithm uses the GNU Scientific Library (GSL), which is a collection of routines for solving computing problems concerning linear algebra, statistics, probability, differentiation, integration etc. RANSAC algorithm is used for extracting geometric information between matching keypoints and tries to reject the outliers from the sensory data through an iterative selection of a random subset from the original points. Those are used for obtaining a model which is evaluated for its consistency through the whole original set of points. The process is iteratively executed until it finds a parametric model that best fits the original data points.

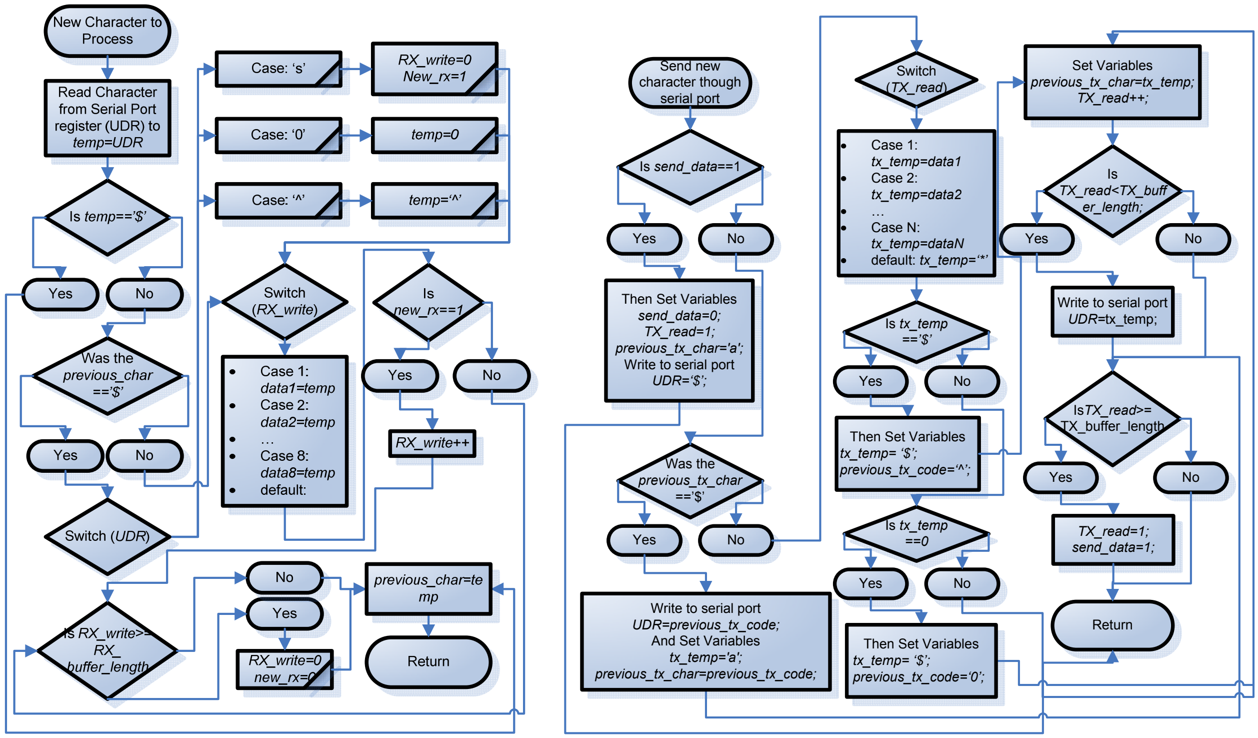

Fig. 3. Special character driven read (left) and write (right) from the serial protocol for both AVR and PC.

Continuing the presentation after our RANSAC detour, the calculation of the SIFT features has a high computational cost, so that in order to have a smooth tracking flow in the object recognition-tracking task, between two consecutively SIFT identifications, we use the Lucas Kanade (LK) optical flow algorithm [19], for tracking the keypoints from the last identified object. The program is able to search for only one specific pattern each time and can be programmed to search in a rotary basis the patterns we want. The pattern we search can be changed dynamically by the GUI for the user itself or automatically by MATLAB responsible for the flying of the UAV system.

Searching for SIFT features within a captured frame is time consuming. Thus, in order not to have discontinuity in the object recognition-tracking task, we have used the LK optical flow algorithm [19], for tracking the SIFT keypoints from the captured frame, which were identified as common with the pattern, in correlation with the perspective transformation of the pattern's edges (top_left, top_right, bottom_left and bottom_right) on the captured frame. When we have a successful match from the execution of RANSAC on the matched keypoints between the captured frame and the pattern, we store for each keypoint its coordinates and its absolute distance between all four edges (top_left, top_right, bottom_left and bottom_right). When a new captured frame arrives, the LK algorithm locates the new coordinates for the matched keypoints based on the last known position that we have stored. Utilizing the known edges of the pattern at the grabbed frame the program does the opposite action from the previous act; it calculates the new edges using the absolute distance from the edges from the previous keypoints and the ones tracked the LK algorithm. It then recalculates the absolute distance between each tracked keypoint and the freshly calculated edges. This is illustrated on a flowchart at Fig. 5. For the LK task, we used the available LK implementation provided by OpenCV.

The routine responsible for finding the SIFT features on the captured frame is executed on a different thread than the one running the main loop of the programme. Until the aforementioned thread is executed, we use the tracked points from the LK in order to determine the position of the edges of the pattern on the captured frame (Fig. 5).

In order to determine how an unknown environment resembles to known patterns, the captured frame is segmented into small overlapping squares of pixels, which are correlated along all the patterns from the histogram pool. The pattern with the highest correlation is assigned to that specific square. By detecting blobs we might obtain regions of interest for further processing. These regions could signal the presence of objects or parts of objects in the captured frame. For example, a blob of patterns from local area's trees that inside contain a blob of patterns from tarmac could represent a potential site for emergency landing.

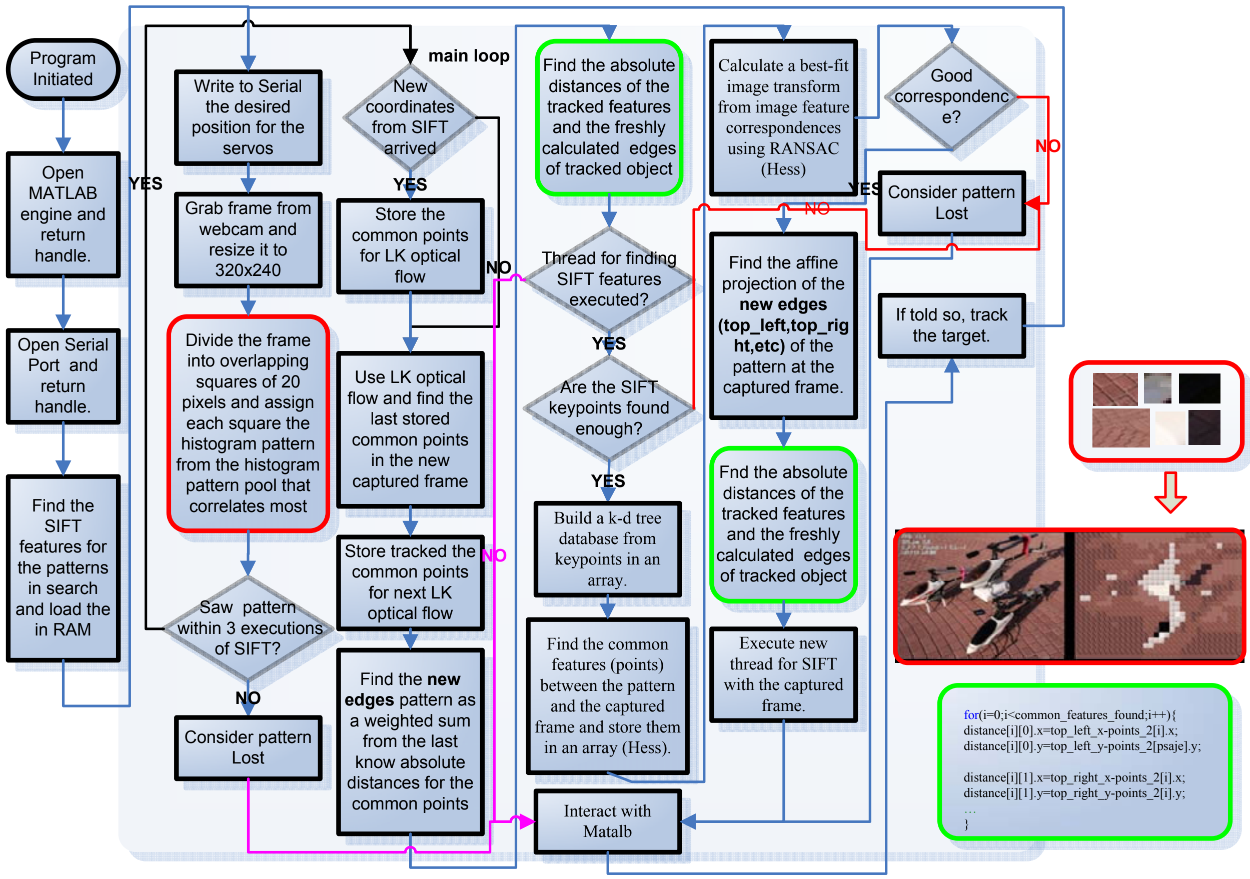

Fig. 4. Flow chart of the program.

Finally, the camera system can track a locked pattern utilizing its pan-tilt capabilities. The program interacts with Matlab, though the Matlab engine. The program writes the fused data into the Matlab's workspace and read the commands, in exactly the same manner, the user of the program would manually do.

As mentioned earlier similar provision for machine vision has also been built in [2] and [3], using the same software we did as a basis for developing the vision system. They have harnessed also SIFT for object recognition on UAVs. Their motivation is to incorporate vision systems in UAVs in order to increase their navigation capabilities. They state that they desire to control the position of the helicopter based on visual data or in a characteristics extracted from the image. Also as they assert, their system could be possibly used for hovering of the helicopter and track moving objects using their Pan –Tilt platform. There are two primary differences in our implementations. Firstly, their tracking system, which relies on SIFT for recognition and is computational demanding (it is based upon the same implementation [11]), is not clarified if it can work on a real time basis like ours.

Secondly, they have adopted a scheme for local processing, in order to increase their speed. This scheme is to select an area of interest in their image for processing. If they don't have a match they increase their area of interest and they process the bigger area. Their approach is interesting and in combination with ours could be very fruitful.

IV. Target Tracking

One simple operation for our vision system is target tracking taking advantage the pan-tilt capabilities of our system. The tracking routine embedded into the C++ program is very in simple but has the potentials to become more ingenious through a fuzzy controller we are planning to built, that will move the camera in a more predictive fashion, using the fused information from the visual system and the inertial data from the helicopter. A pseudo code of the routine is shown at Fig. 6. The aim of this routine is to keep the estimated center of the tracked object within a specified *margin* from the center of the captured image. Something that we discovered after experimenting is that it is better to change the camera's position as little as possible in order to move the camera towards the estimated center for the pattern. This is because, in abrupt change of position, the LK kanade algorithm wasn't able to keep in track the keypoints, so the target would be lost either way.

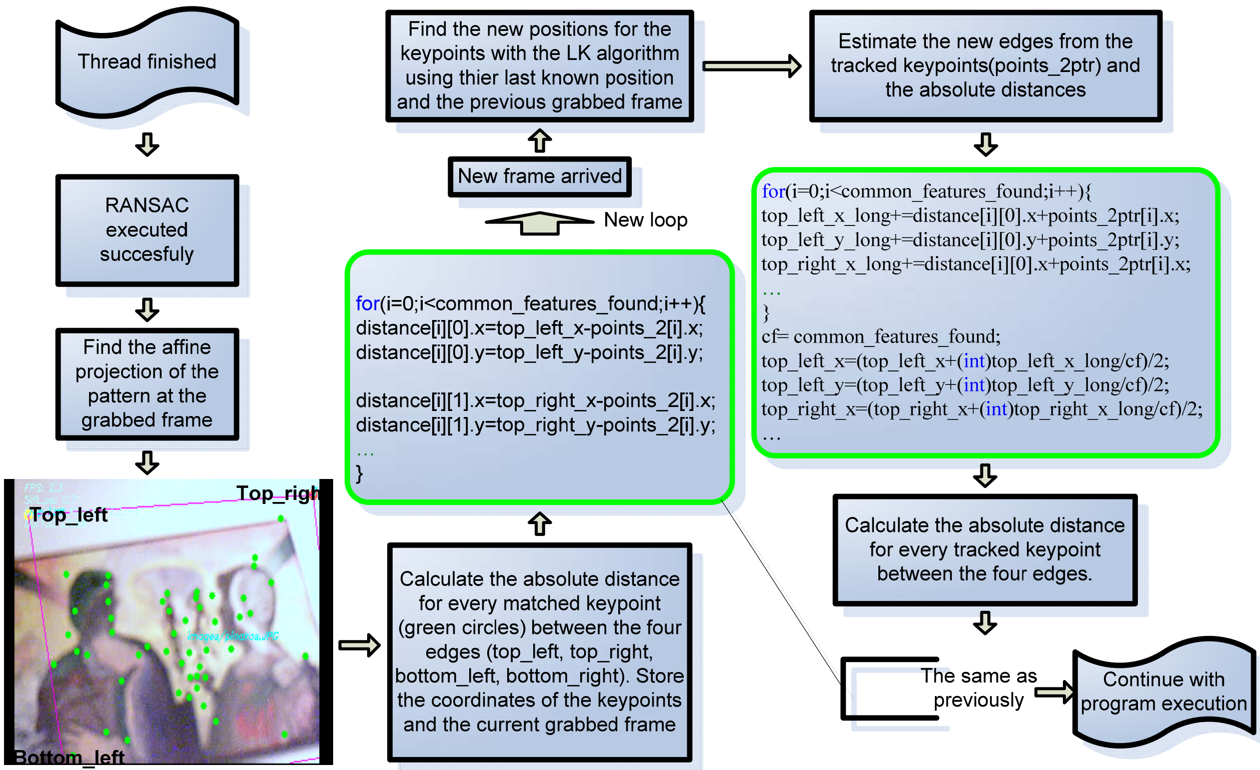

Fig. 5. Lucas Kanade optical flows for smooth tracking between consecutive SIFT recognitions.

```
margin=20;
position=image_center_x-(pic_width/2);
if(fabs(position)> margin){
   if(position>0)
      camera_left_right=camera_left_right+1;
   else
      camera_left_right=camera_left_right-1;
 }
position=image_center_y-(pic_height/2);
if(fabs(position)> margin){
   if(position>0)
      camera_up_down=camera_up_down-1;
   else
      camera_up_down=camera_up_down+1;
 }
```

Fig. 6. Track target.

An application of our implementation is shown in Fig. 7. Four snapshots are presented from a test flight. The helicopter takes off and flies across a building surrounded by plain nature. The image of the black door of the building is inserted to our software in order to be used as the reference image. As it can be seen in the snapshots software recognizes in real time the black door and highlights it (pink rectangle), while the helicopter flies around the building. The coordinates of the tracked object in the image pass in the Matlab engine in order to be used in the navigation of the helicopter. These coordinates are also used to appropriately guide the pan-tilt mechanism in order to continuously track the object while the helicopter flies and the tracked image is still inside the picture.

Object tracking looks to work very well in this test flight, although we noticed a lot of vibration in the camera, becoming from the helicopter engine. Calculations and object tracking work really fast.

V. Conclusion

A vision system to aid the navigation of an unmanned helicopter is presented in this work. This system, including a web camera mounted on a custom made pan tilt mechanism, is loaded on an unmanned helicopter and software has been implemented, based on Open Computer Vision Library (OpenCV), for various applications such as real time object recognition, pattern matching and object tracking. This system interacts with the helicopter control system using the Matlab engine, aiding helicopter's navigation. Preliminary test flights show that this system is promising and can be developed in order to assist the helicopter in performing various applications.

Fig. 7. Real time tracking onboard helicopter

REFERENCES

[1] David G. Lowe, "Distinctive image features from scale-invariant keypoints," *International Journal of Computer Vision,* vol. 60, no. 2, pp. 91-110, 2004.

[2] I.F. Mondragon, P. Campoy, J.F. Correa, L.Mejias, “Visual Model Feature Tracking for UAV Control,” in *IEEE International Symposium on Intelligent Signal Processing*, 2007.

[3] Pascual Campoy, Juan F. Correa, Ivan Mondragón, Carol Martínez, Miguel Olivares, Luis Mejías, Jorge Artied, “Computer Vision Onboard UAVs for Civilian Tasks,” *Journal of Intelligent and Robotic Systems*, vol. 54, no. 1-3, pp. 105-135, 2009.

[4] Yuping Lin, Qian Yu, Gerard Medioni, “Map-Enhanced UAV Image Sequence Registration,” in *Proceedings of the 8th IEEE Workshop on Applications of Computer Vision*, 2007.

[5] Wolfgang Forstner and Richard Steffen, “Online Geocoding and Evaluation of Large Scale Imagery Without GPS,” in *Photogrammetric Week*, 2007.

[6] Todd Templeton, David H. Shim, Christopher Geyer, Shankar S. Sastry, “Autonomous Vision-based Landing and Terrain Mapping Using an MPC-controlled Unmanned Rotorcraft,” in *IEEE International Conference on Robotics and Automation*, 2007.

[7] Wei Zhang and Jana Kosecka “A new inlier identification scheme for robust estimation problems,” in *Robotics: Science and Systems II*, 2006.

[8] Samer M. Abdallah, Daniel C. Asmar and John S. Zelek, “A Benchmark for Outdoor Vision SLAM Systems,” *Journal of Field Robotics*, vol. 24, no. 1-2, pp. 145-165, 2007.

[9] Herbert Bay, Tinne Tuytelaars, Luc Van Gool, "SURF: Speeded Up Robust Features", in *Proceedings of the 9th European Conference on Computer Vision*, May 2006.

[10] *SURF*. Available: http://www.vision.ee.ethz.ch/~surf/

[11] *Robb Hess*, http://web.engr.oregonstate.edu/~hess/

[12] Nikos I. Vitzilaios and Nikos C. Tsourveloudis, “An Experimental Test Bed for Small Unmanned Helicopters,” *Journal of Intelligent and Robotic Systems*, vol. 54, no. 5, pp. 769-794, 2009.

[13] *FTDI*, Future Technologies International, http://www.ftdichip.com/

[14] *Open Computer Vision Library (OpenCV)*. Available: http://opencvlibrary.sourceforge.net/

[15] Gady Agam, ”Introduction to programming with OpenCV”, 2006. Available: http://www.cs.iit.edu/~agam/cs512/lect-notes/opencv-intro/index.html

[16] David G. Lowe, "Object recognition from local scale-invariant features," in *International Conference on Computer Vision,* pp. 1150-1157, Corfu, Greece, 1999.

[17] Jeff Beis and David G. Lowe, “Shape indexing using approximate nearest-neighbour search in high-dimensional spaces,” in *Conference on Computer Vision and Pattern Recognition*, pp. 1000–1006, Puerto Rico, 1997.

[18] Martin A. Fischler and Robert C. Bolles, “Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography,” *Communications of the ACM*, vol. 24, no. 6, pp. 381-395, 1981.

[19] B.D. Lucas and T. Kanade, “An iterative image registration technique with an application to stereo vision,” in *Proc. of the 7th IJCAI*, pp. 674–679, Vancouver, Canada, 1981.

[20] *GNU Scientific Library (GSL)*, Available: http://gnuwin32.sourceforge.net/packages/gsl.htm