ANALYZING UAV MISSIONS IN A SCALED ENVIRONMENT: EXPERIMENTAL RESULTS

Keith W. Sevcik, James T. Hing, Paul Y. Oh
Drexel Autonomous Systems Lab
Drexel University
Philadelphia, PA, US
email: kws23, jth23, pyo22 @drexel.edu

ABSTRACT
UAV research has grown significantly over the past 10 years, developing well defined areas of study with many active research efforts. More often than not, a single objective has been investigated with different platforms, control algorithms and sensors. To judge and compare these different solutions, there must be a common testing environment. However, logistical issues make this almost impossible, giving little or no basis for comparison. This paper presents a testing facility that can be quickly outfitted with different aircraft models, control algorithms and sensors. A 6 DOF gantry surrounding a scaled urban environment allows for control algorithms to be tested against real sensor data. The environmental conditions in the facility can be adjusted in a controlled, repeatable manner. Results from scaled tests are shown to match full scale computer simulation. Also, preliminary tests show how the facility can be used to test varied environmental conditions.

KEY WORDS
Robotics, Unmanned and Underwater Vehicles

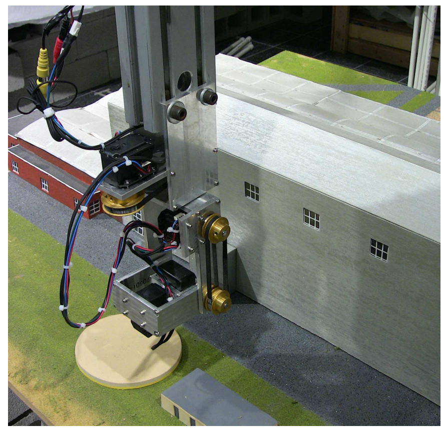

Figure 1. The UAV sensor package is mounted to the end effector of a 6 DOF gantry. Using real sensor data, the control algorithm and math model of the UAV, closed loop control can be emulated in a scaled environment.

1 Introduction

In the past 20 years UAV research has grown from a relatively non-existent field to a major branch of robotics. This increase in interest has identified many fundamental problems and core research areas. However, despite a plenitude of research and proven solutions, many of these problems remain ambiguously open.

For instance, necessary components such as collision avoidance have many robust and complete solutions. [1] utilized optical flow and stereo vision to guide a helicopter through urban canyons. [2] also achieved autonomous navigation through an urban environment using a scanning laser range finder. These two works achieve the same goal. However, differences in design and experimentation make it difficult to choose the best solution.

Another thoroughly investigated problem is navigating towards a target. [3] used a KLT tracker to servo a helicopter towards a window. [4] guided a helicopter to a landing pad using template tracking. These lines of research were fundamentally similar, the major difference being the algorithm. Again it is difficult to compare results because the environment and testing conditions were not the same.

Research has been done to create a common simulation environment for UAVs. [5] created a computer simulator for testing different UAVs and missions. These computer simulations work for designing flight controllers but they fail to address implementation issues. Environmental effects such as rain, dust and smoke are difficult to model. Furthermore, hardware problems such as noisy data or communication lags are often overlooked in simulation.

Work has been done to incorporate UAV hardware. This usually involves interfacing the avionics with a simulator as in [6]. This approach is effective at testing autopilots and can approximate performance of control algorithms that use simple sensors such as GPS. However, it falls short of a fully integrated test with mission sensors such as cameras.

Other research has gone further to include UAV hardware in simulation. [7] presents an effort to incorporate actual IMU data. [8] performed a bench top test of the entire UAV hardware configuration. Like wise, [9] sought to characterize mission sensor performance. The components exist to create a UAV mission testing facility, but to date there is no comprehensive solution for comparing different approaches to UAV missions.

To compare algorithms, sensors and platforms, there must be a common, fixed testing environment. Tests must be run on the real hardware to capture effects like data lag, sensor noise and hardware/software glitches. Flying the vehicle can be arduous and dangerous, but in absence of a real flight, the dynamics must be accurately approximated. Finally, there should be a method to characterize the solution based on its robustness to environmental conditions.

This paper presents the design of such a testing facility and its use to test UAV missions. The facility is comprised of a scaled environment and the Systems Integrated Sensor Test Rig (SISTR). SISTR utilizes the UAV sensors mounted on a 6 DOF gantry to perform closed loop control of the UAV in a realistic environment. The UAV's control algorithm generates control commands from real sensor data. These commands are provided to the mathematical model of the UAV, which outputs translational and rotational positions. This data is then used to emulate flying the sensors through the scaled mission environment, as shown in Fig. 1.

The use of scaled models in flight simulation is not new. As described in [10], early flight simulators used scaled models of terrain to provide visual feedback for pilots. These systems provided high fidelity, realistic visual cues for pilots. However, flights were limited to the area of the models. This approach was abandoned in favor of computer simulation which provided endless terrain at the sacrifice of realism.

The problem faced simulating UAV missions is the opposite. Missions are confined to a defined region such as a town. Computer simulations try to model real world effects but fail to capture the caveats of these environments. Changes in lighting, fog and dust are difficult to model in simulation. Through work on a Class II UAV for Future Combat Systems, the authors have gained experience recreating these conditions and characterizing their effects on LIDAR, sonar, UWB radar and other sensors. This approach to testing sensors and control algorithms provides deeper insight into the performance of the UAV.

This paper is organized as follows: Section 2 describes the testing facility, how it approximates flight and incorporates actual hardware. Section 3 describes tests to show how well aircraft dynamics can be approximated at scale. Section 4 looks into issues encountered when scaling sensors. Results from scaled tests are compared to results from computer simulation in Section 5. Section 6 shows how the facility can be used to measure the effects of obscurants and varied illumination. Finally, Section 7 forms conclusions and Section 8 outlines future work.

2 Concept

Our realization of a UAV mission testing facility contains 3 major components: a scaled environment, a 6 DOF gantry, and a mathematical model of the UAV. The scaled environment provides the test space where missions are conducted.

Figure 2. The Piasecki Aircraft research facility represents typical terrain for UAV missions, as shown in this satellite image. The urban and wooded environments contain buildings, trees, poles and thin wires. The area spans several hundred meters, allowing ample room for UAV flight tests.

Figure 3. The 1/87th scale environment was built to visually approximate the full scale facility. Major man-made features were carefully measured and scaled. Textures and colors were visually matched to facilitate testing of vision sensors. Natural terrain was roughly approximated with modeling supplies.

The mathematical model is used to generate the motions of the aircraft based on control commands. These motions are then replayed in real time on the 6 DOF gantry. These components are described in detail below.

2.1 Scaled Environment

The scaled environment was designed to accommodate the typical UAV mission. The types of missions we're investigating are those executed on-station such as reconnaissance, perch-and-stare and payload delivery. These missions often involve fundamental capabilities such as autonomous landing, mapping, and target tracking.

Since many UAV missions occur amongst natural and man made features, it was necessary to incorporate both in the test space. Obstacles such as buildings, trees and poles

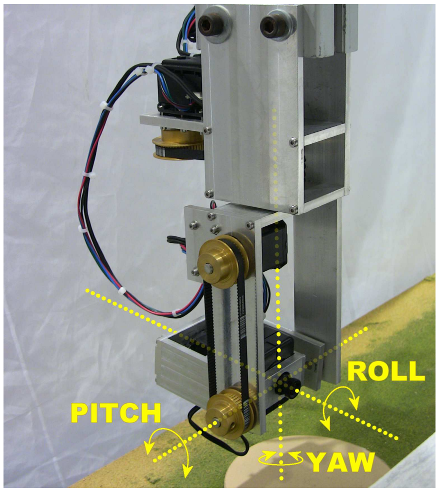

Figure 4. A custom pan/tilt/roll unit was created such that all axes of rotation intersected at the center of the sensor package. It was also made to fit a small form factor so that it could move very close to the scaled environment.

are typical of these environments. Piasecki Aircraft's research facility shown in Fig. 2 provides a mix of man-made and natural features. It also contains challenging obstacles like thin wires and branches. This makes Piasecki a versatile and representative test site.

We looked to previous research to determine the size of the testing area. In both [2] and [11] the operational area was as large as $220m$ x $220m$ flown at altitudes in the 10's of meters. This suggests that an appropriate area for testing is $300m$ x $300m$. The work space inside the 6DOF gantry is $5.40m$ x $4.22m$ x $1.65m$. From these constraints, we chose to model the environment at 1/87th scale, corresponding to the common H0 modeling scale.

Emphasis was placed on replicating visual features, as vision is a common UAV sensor and plausible to scale. The buildings and other man made structures were created from foam core board. The major dimensions and features were measured and scaled. Textures and colors were made to visually approximate the actual surfaces. Natural features were roughly approximated using modeling materials. The resulting model is pictured in Fig. 3.

2.2 SISTR

The scaled model is enclosed within the Systems Integrated Sensor Test Rig (SISTR), described in detail in [12]. SISTR is a UAV testing facility supported by the U.S. National Science Foundation that can mimic UAV flight in varied environmental conditions. Motion is achieved with a 6 DOF gantry that spans the working volume. The translational motions are achieved with large DC motors driving a vertically mounted beam. The remaining 3 DOF are provided by a pan/tilt/roll unit attached to the end of this beam, together forming the end effector of SISTR.

Table 1. CONSTRAINT VELOCITIES

Axis	Gantry	Scaled
X	0.012 - 0.61m/s	1.04 - 53.0m/s
Y	0.019 - 0.61m/s	1.65 - 53.0m/s
Z	0.021 - 0.61m/s	1.83 - 53.0m/s

There were several design constraints for the pan/tilt/roll unit. First, the axes of rotation must intersect at the center of the sensor, mimicking how most aircraft rotate near their center. This also decouples rotations, allowing independent control over each axis. The pan/tilt/roll unit must also be small. Since the end effector takes up space in the scaled model, it creates physical constraints on what can be simulated. For instance, the width of the end effector limits how narrow of an urban canyon can be traversed. This drove the end effector to be as small as possible.

The pan/tilt/roll unit shown in Fig. 4 was constructed from Dynamixel AX-12 servos because of their small size, high torque, high speed and resolution. The servos can be controlled to $0.35\deg$ at rates up to $300\deg/sec$. Since rotations do not scale, this corresponds to $0.35\deg$ and $300\deg/sec$ at full scale.

Another design challenge in scaling is matching the velocities and positions of the UAV. Table 1 displays the maximum and minimum velocities of the gantry and their scaled equivalents. While these do not represent the full range of UAV velocities, they do encompass a large portion of the operating range. All translational axes can be controlled to within $\pm 0.5cm$. This scales up to a resolution of $\pm 0.43m$. This is well within the $\pm 2m$ accuracy of off the shelf GPS systems.

Sensors mounted on SISTR can be virtually flown through an environment. Real time data is collected by the software that would be used in flight. Control commands are fed into a mathematical model of the aircraft, which generates aircraft positions to be replayed on SISTR.

SISTR also has testing apparatus to simulate different environmental conditions. Some fixtures are permanent. Stage lights can be individually controlled to create varied lighting scenarios. Light-blocking curtains can be used to create night time conditions. Other environmental fixtures can be added as need. A fog generator has been used to simulate obscurrants. A rain and dust machine [9] were also created to simulate more extreme operating conditions.

SISTR is programmed to move to a commanded position. These commands are sent from a remote computer workstation through UDP communication. Positions are generated by the mathematical model of the UAV, which is implemented in the flight simulator software X-Plane.

2.3 X-Plane

X-Plane is a popular, widely available flight simulator made by Laminar research. X-Plane models dynamics

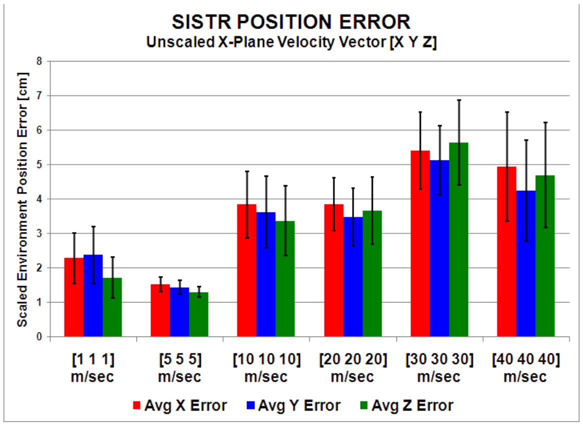

Figure 5. Tests were conducted to see how well SISTR followed motions commanded from X-Plane. While SISTR is capable of positioning within $\pm 0.5cm$, there is lag between the commanded and actual position. The lag was smallest at $5m/sec$. Below, $30m/sec$ the lag between command and motion is within a workable range.

based on the geometry of the aircraft. Using blade element theory, the craft is divided into smaller elements. The lift and drag forces are summed to determine the motion of the entire aircraft. X-Plane can be used to model many different aircraft, from planes to helicopters to rockets. While X-Plane models do not capture the quirks of every aircraft, the general behavior is closely approximated.

X-Plane is a versatile research tool because it is highly customizable. The user can create custom aircraft, environments, weather and lighting conditions. It is also possible to interface with the program through UDP. This allows the user to obtain program data and build their own plug-ins.

X-Plane has been validated as a design tool by many other research efforts. For instance [13], utilized X-Plane to model a Maxi/Joker rotorcraft. A fuzzy logic controller for a rotorcraft was tested using X-Plane in [14]. [15] discusses how UAV pilots can be trained using X-Plane. Moreover, X-Plane has been FAA certified for flight training.

We are using X-Plane to simulate the flight dynamics of a rotorcraft. Control commands are sent to X-Plane rotorcraft model. X-Plane then generates the motions of the helicopter and reports them back to SISTR. To achieve this, a custom UDP plug-in was programmed that communicates with X-Plane at up to 90 Hz.

This system allows UAV flight to be mimicked on a small scale. For this implementation to work, several practical concerns must be addressed. Two predominant concerns that must be addressed are the ability of SISTR to mimic aircraft dynamics and the scalability of sensors.

3 Scalability of Aircraft Dynamics

There are several factors that come into play when attempting to replicate the motions of an aircraft in a scaled model.

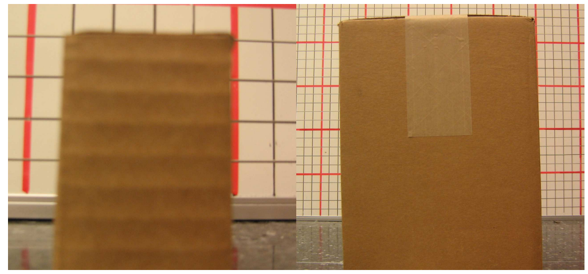

Figure 6. Pictures of a 1/5th scale box in front of a grid (left) and a full scale box in front of the same grid (right). The distance from the grid to box and grid to camera were scaled accordingly. Despite scaling, the field of view and relative size of the box do not change.

the first question that arises is how the aircraft dynamics are affected by scaling. In our solution, X-Plane simulates the motion of the aircraft at full-scale. The resulting position of the aircraft calculated by X-Plane is then scaled and used to command SISTR. The question the becomes how well does the hardware follow the positions generated by the mathematical model?

This is greatly affected by the update rate. The faster the communication between X-Plane and SISTR, the finer the motion command. This directly effects the range of speeds that can be accurately reproduced. Faster speeds demand more position commands per second.

To test this range, aircraft were flown in X-Plane at different velocities. The velocity vectors applied ranged from $\begin{bmatrix} 1 & 1 & 1 \end{bmatrix} m/sec$ to $\begin{bmatrix} 40 & 40 & 40 \end{bmatrix} m/sec$, corresponding to speeds ranging from $1.73m/sec$ to $69.28m/sec$. The velocities were gradually varied to determine if there was an upper or lower bound on the velocities SISTR could reproduce. The direction was chosen so that all 3 axes were equally actuated.

The errors in positioning are shown in Fig. 5. While SISTR is capable of positioning within $\pm 0.5cm$, there is lag between the commanded and actual position. When the aircraft was moving slowly at $\begin{bmatrix} 1 & 1 & 1 \end{bmatrix} m/sec$, the motion of SISTR was very rough. This is because the change in commanded position is very small. It is difficult to overcome static friction in very small increments. When the aircraft was moving at velocities of $\begin{bmatrix} 30 & 30 & 30 \end{bmatrix} m/sec$ or greater the motion was very erratic. This is due to reaching an upper limit on the motion of SISTR. The best results were obtained at a velocity of $\begin{bmatrix} 5 & 5 & 5 \end{bmatrix} m/sec$.

4 Scalability of Sensors

Scaling sensors presented itself as the first obvious limitation of using scaled models. When choosing missions to evaluate, the sensor package must be conducive to scaling. Foq example, sonar does not scale well. Since sound decays exponentially over distance, detection cones are large,

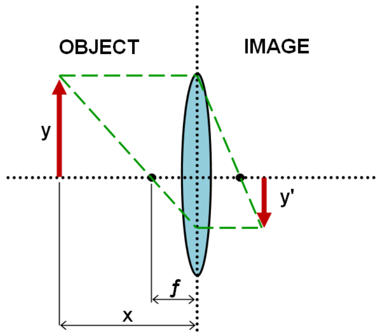

Figure 7. For evaluating how a camera scales, the thin lens approximation is used. An object y a distance x from the lens with focal point f is projected into the image as y'.

and materials reflect sound differently, performing scaled sonar tests would require a custom scaled sensor.

However, other sensors such as vision sensors scale more intuitively. The simplest model of a vision sensor - a pinhole camera - is obviously scalable. Compound lens cameras present more issues, but these extremes bound a range of cameras that can be modeled at smaller scales.

We first looked at how fundamental camera parameters scaled. One important parameter is the focal length. The focal length of a camera determines its field of view, depth of field, magnification and many other properties. To determine how focal length scaled, we compared two objects projected onto the image plane. Assuming a thin lens as shown in Fig. 7, an object of length y has length y' when projected by a lens with magnification M:

$$y' = My \tag{1}$$

Assuming a thin lens, the magnification of an object depends on the focal length f and the distance to the object x:

$$M = \frac{f}{f - x} \tag{2}$$

Then, the size of an object in the image plane and an object scaled by a factor n is:

$$y' = (\frac{f}{f - x})y \tag{3}$$

$$y'_s = (\frac{f_s}{f_s - nx})ny \tag{4}$$

The goal is to find an equation for the focal length of the scaled camera f_s such that the size of the object in the image plane does not change, in other words $y' = y'_s$. Setting the two equations above equal to each other and solving for f_s yields:

$$f_s = \frac{nfx}{nx + f(1 - n)} \tag{5}$$

If $x >> f$, the above simplifies to:

$$f_s = f \tag{6}$$

In summary, using the thin lens assumption and assuming that the distance to the object is much greater than the focal length, the focal length of the camera is not affected by scaling. Consequently, the angular field of view of the camera α also remains unaffected by scaling:

$$\alpha = 2 \arctan(D/2f) \tag{7}$$

Where D is the dimension (width or height) of the image plane. This seems counter intuitive. However, this fact has been exploited by film makers substituting miniature props for life size objects. Fig. 6 shows an example of this effect. A box was photographed at a fixed distance from a grid. A 1/5th scale box was then photographed in front of the same scaled grid, adjusting the distances proportionally. Counting the grid squares in the field of view, it can be seen that scaling does not effect the relative size of the object or field of view of the camera.

One problem with scaling is that the objects must be closer to the camera. If the object moves out of the depth of field it will no longer be in focus. Fortunately, the camera can be refocused without drastically changing results.

To test how scaling effects a common vision algorithm, a Kanade-Lucas-Tomasi (KLT) feature tracker was applied to video from a scaled model and a real world building. As shown in Fig. 8, the edges of the building and

Figure 8. A Kanade-Lucas-Tomasi feature tracker was applied to video from the scaled model and the real world. Similar features were detected and tracked in both data sets. This confirms that vision sensors produce similar results at both small and full scale.

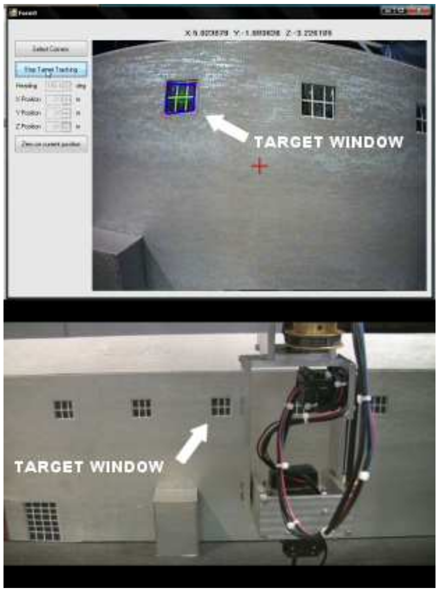

Figure 9. Visual servoing tests were conducted in both simulation and in the scaled environment. This figure shows the camera positioned in the scaled environment. A program was created to autonomously servo the camera such that the target window was in the center of the visual field.

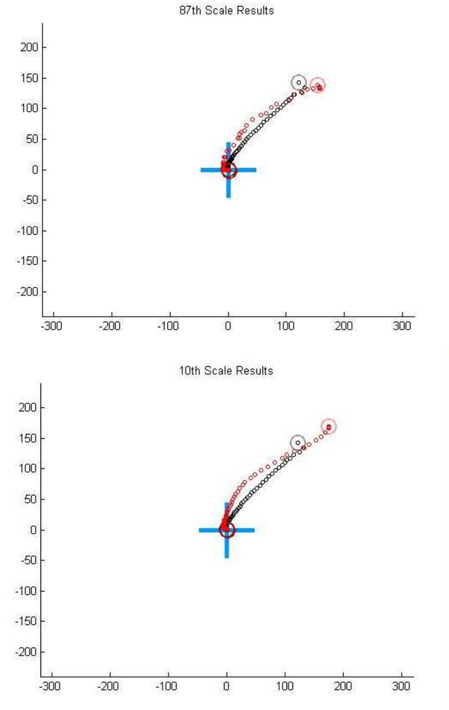

Figure 10. These results show the position of the window in the visual field over time. The black circles show data from simulation, the red from scaled model tests. The large circles mark the start and end points of the test, and the plus sign marks the center of the visual field. At 1/87th scale, the motion was similar to the baseline, though jerky due to static friction. At 1/10th scale, the effect of static friction was not noticeable.

corners of the window were selected as the strongest features to be tracked. These were detected in both the scaled model and the full scale facility. This confirms that the algorithm was indifferent to the scale of the facility.

5 Scaled Tests vs. Simulation

The concept was further verified against baseline computer simulations. Computer simulation serves as a sufficient approximation of real life flight. Moreover, the conditions in a computer simulation are constant, providing a consistent basis for comparison.

A standard UAV mission was needed to conduct this comparison. Visual servoing is a common task performed by UAVs. It has many applications, from autonomous landing to obstacle avoidance. The work of [3] is representative of a helicopter visual servoing task. This work was recreated in simulation and in the scaled environment. The method is presented here in brief.

In [3], tests were performed with the fully autonomous COLIBRI helicopter outfitted with a forward facing camera. To mimic COLIBRI, an attitude, position, and velocity controller were programmed to control the X-Plane model. The attitude controller performed PID control of the cyclic pitch, cyclic roll, collective and tail rotor to stabilize the helicopter at a given yaw, pitch, roll, and altitude. The position and velocity controllers performed PID control of the pitch, roll, and altitude to move the helicopter to a given position or at a given velocity. This made the X-Plane helicopter functionally identical to COLIBRI.

The goal of [3] was to select a window and have the helicopter guide itself to the center of the window while remaining a fixed distance from the building. This was accomplished by measuring the distance from the center of the window to the center of the visual field. The velocity of the aircraft was proportionally adjusted to this error using the following equations:

$$vy_ref = 2\frac{i - w/2}{w} \tag{8}$$

$$vz_ref = 2\frac{j - h/2}{h} \tag{9}$$

where vy_ref and vz_ref are the horizontal and vertical velocities, i and j are the image coordinates of the center of the target window, and w and h are the width and height of the image. This algorithm was implemented on the X-Plane helicopter, allowing it to be visually servoed when connected to a video feed.

For the computer simulation the video came from a screen capture of the X-Plane environment. The Piasecki test site was modeled in X-Plane. Textures were applied to visually match the facility. For scaled tests, the video came

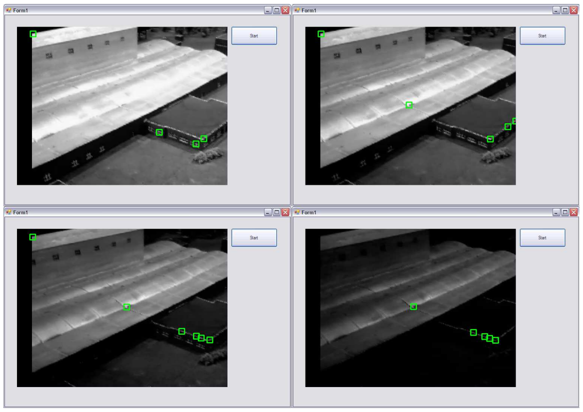

Figure 11. The KLT tracker was applied to video taken at illuminations ranging from an overcast day to sunset. As the scene got darker, features such as the window were lost. The prominent white edge of the building became the strongest feature to track. Features in the darker scenes were generally weaker and more difficult to track.

Figure 12. KLT tracking was also performed in an environment obscured with fog. The image became more washed-out, but the same features were still detected. These features were weaker than those found in the unobscured image, making them more difficult to track.

from a camera in the scaled environment. The only difference between the two tests was the source of the video.

Several scenarios were conducted in simulation and the scaled environment. The initial position was changed for each scenario. The window was positioned at the edges and corners of the visual field for a total of 8 scenarios. Figure 9 shows the camera image and position when the window was in the upper left side of the image.

Results showed that tests at 1/87th scale were similar to simulated tests. During the experiments, it was noticed that friction in SISTR effected the results. At 1/87th scale, the motions of SISTR were so small that sometimes they didn't overcome static friction. To verify this, tests were run at 1/10th scale. Both results are shown in Figure 10. As can be seen, the motions of SISTR matched more closely when not limited by static friction.

6 Illumination/Obscurants Tests

After affirming scaled tests, experiments were conducted to see how illumination and obscurants effect the KLT tracker. First illumination was varied. The baseline illumination was $3,020 lux$, corresponding to an overcast day. Tests were run at illuminations of $2,160 lux$, $1,340 lux$ and $480 lux$, the latter corresponding to sunset.

Fig. 11 shows the results. As illumination decreases, features around the windows of the building are lost. Instead, the white border around the top of the building is favored. Also, a shadow on the roof becomes more prominent as the lighting decreases, causing a feature to be detected on the roof. Generally, decreased illumination resulted in weaker and more difficult to track features.

After testing the effect of lighting, fog was introduced. As can be seen in Fig. 12, fog tends to wash-out the image, but the same features are detected. However, these features are not as strong and are more difficult to track.

7 Conclusions

Results have shown bounds on the ability to recreate scaled missions. Motion testing showed that velocities were limited between $\begin{bmatrix} 1 & 1 & 1 \end{bmatrix} m/sec$ and $\begin{bmatrix} 20 & 20 & 20 \end{bmatrix} m/sec$, corresponding to speeds of $1.73 m/sec$ and $34.64 m/sec$. This encompasses the typical cruise speed of smaller UAVs. The performance was best at a velocity vector of $\begin{bmatrix} 5 & 5 & 5 \end{bmatrix} m/sec$. This suggests that slower moving UAVs such as rotorcraft would be easily represented. Issues of position error could be remedied by increasing the scale. This becomes a balance between modeling the aircraft and modeling the environment. The solution would be problem specific.

Similarly, the effects of static friction in the gantry are magnified at smaller scales. These effects can be allayed by increasing the scale. This could also be overcome by improving SISTR's control software. By incorporating a friction model, the effects of friction could be eliminated.

As concluded earlier, not all sensors are scalable. There are some physical limitations for scalable sensors such as vision. When objects scale they move closer to the camera, going out of focus. If the objects move too close it may not be possible to bring them into focus. Size constraints are another issue. The size of cameras and actuators puts physical limits on how close the sensor can approach the ground or obstacles. For these reasons, scaling is more applicable to problems with distance between the UAV and its surroundings. Again, these issues can be addressed by increasing the scale.

Interesting effects were noticed from varying the environmental conditions. Though fog and weak illumination resulted in weaker features, vision still worked under these conditions. This shows promise for developing algorithms that operate under adverse conditions. These results show scaling to be a promising avenue for designing UAV missions. The authors intend to further investigate its potential, as described below.

8 Future Work

Now that it has been confirmed that motions can be recreated and that vision is scalable, the authors seek to characterize solutions to UAV missions. Target tracking is one key area where a baseline for comparison is needed. Future work would recreate seminal target tracking research in the scaled model. Performance metrics will be designed and tested against.

After establishing a baseline for target tracking, the authors seek to address other fundamental research such as mapping and obstacle avoidance. The end goal is to produce a set of baseline results from which others can compare their solutions.

9 Acknowledgements

The authors wish to thank Jesse Greenberg of Simulab Studios for the construction of the scaled testing environment. Thanks also go to Piasecki Aircraft for their continued support and use of their testing facilities.

References

[1] S. Hrabar *et al.*, "Combined optic-flow and stereo-based navigation of urban canyons for a uav," in *International Conference on Intelligent Robots and Systems (IROS)*, Edmonton, AB, Canada, Aug. 2005, pp. 3309–3316.

[2] S. Scherer *et al.*, "Flying fast and low among obstacles," in *International Conference on Robotics and Automation (ICRA)*, Rome, Italy, Apr. 2007, pp. 2023–2029.

[3] L. Mejias *et al.*, "Visual servoing of an autonomous helicopter in urban areas using feature tracking," *Journal of Field Robotics*, vol. 23, no. 3–4, pp. 185–199, Apr. 2006.

[4] S. Saripalli *et al.*, "Vision-based autonomous landing of an unmanned aerial vehicle," in *Int. Conf. on Robotics and Automation (ICRA)*, Washington, DC, May 2002, pp. 2799–2804.

[5] J. Nalepka and M. Duquette, "A multi-purpose simulation environment for uav research," in *AIAA Modeling and Simulation Technologies Conference and Exhibit*, Austin, TX, Aug. 2003.

[6] E. R. Mueller, "Hardware-in-the-loop simulation design for evaluation of unmanned aerial vehicle control systems," in *AIAA Modeling and Simulation Technologies Conference and Exhibit*, Hilton Head, SC, Aug. 2007.

[7] S. Xianjun *et al.*, "Hardware-in-the-loop simulation framework design for a uav embedded control system," in *Chinese Control Conference*, Harbin, China, Aug. 2006, pp. 1890–1894.

[8] A. H. Goktoian *et al.*, "Real time multi-uav simulator," in *Int. Conf. on Robotics and Automation (ICRA)*, Taipei, Taiwan, Sep. 2003, pp. 2720–2726.

[9] K. W. Sevcik and P. Y. Oh, "Designing aerial robot sensor suites to account for obscurants," in *International Conference on Intelligent Robots and Systems (IROS)*, San Diego, CA, Oct. 2007, pp. 1582–1587.

[10] D. J. Allerton, "Flight simulation: Past, present and future," *The Aeronautical Journal*, vol. 104, no. 1042, Dec. 2000.

[11] M. A. Hsieh *et al.*, "Adaptive teams of autonomous aerial and ground robots for situational awareness," *Journal of Field Robotics*, vol. 24, no. 11–12, pp. 991–1014, Nov. 2007.

[12] V. Narli and P. Oh, "A hardware-in-the-loop test rig for designing near-earth aerial robotics," in *International Conference on Robotics and Automation (ICRA)*, Orlando, FL, May 2006, pp. 2509–2514.

[13] R. D. Garcia and K. P. Valavanis, "The implementation of an autonomous helicopter testbed," *Journal of Intelligent and Robotic Systems*, vol. 54, no. 1–3, pp. 423–454, Mar. 2009.

[14] B. Vidolov *et al.*, "A two rule based fuzzy logic controller for contrarotating coaxial rotors uav," in *IEEE conference on Fuzzy Systems*, Vancouver, BC, Canada, Jul. 2006, pp. 1563–1569.

[15] J. T. Hing and P. Y. Oh, "Development of an unmanned aerial vehicle piloting system with integrated motion cueing for training and pilot evaluation," *Journal of Intelligent and Robotic Systems*, vol. 54, no. 1–3, pp. 3–19, Mar. 2009.