Intelligent Autonomy and Vision Based Obstacle Avoidance for Unmanned Air Vehicles

Raman Mehra, Jeffrey Byrne and Jovan Bošković
Scientific Systems Company, Inc.
500 West Cummings Park, Suite 3000, Woburn, MA 01801
rkm@ssci.com jbyrne@ssci.com jovan@ssci.com

Abstract— **The paper describes the development and implementation of the Visual Threat Awareness (VISTA) system, its integration with the Multi-layer Architecture for Trajectory Replanning and Intelligent plan eXecution (MATRIX) for autonomous intelligent control of Unmanned Aerial Vehicles (UAV), and performance evaluation of the integrated system through flight tests. The VISTA system generates information on the threats and obstacles in real-time, and passes it on to the MATRIX system that makes mission-related decisions and generates new waypoints and a trajectory that safely avoids the obstacle. The VISTA system combines binocular visual stereo, perceptual organization, graph partitioning and feature tracking for a passive system to enable real-time obstacle detection. Computational stereo performance has progressed such that there now exist several commercial or open source implementations that operate at frame rate, but suffer from well known correspondence errors. We show that introducing a global segmentation step after commodity stereo can increase robustness and leverage existing stereo software. The global segmentation step is based on a graph structure appropriate for collision detection, human vision inspired perceptual organization and graph partitioning using the minimum s-t graph cut. This system has been prototyped using Sarnoff Corp's Acadia I vision processor to enable 640x480@10Hz operation on embedded avionics. We describe VISTA system theory and show proof of concept and flight experiment results of the integrated MATRIX/VISTA system on Georgia Tech's GT-Max autonomous helicopter.**

I. Introduction

UNMANNED Aerial Vehicles (UAVs) are envisioned as an integral part of future military forces. Large scale UAVs will perform autonomous tasks such as high-altitude reconnaissance, Close Air Support, Suppression of Enemy Air Defenses, and aerial refueling. Small scale UAVs will enable on-demand intelligence, surveillance and reconnaissance tasks including: "over the hill" reconnaissance, "perch and stare" surveillance, biological and chemical agent detection, precision strike missions, and battle damage assessment. Such tasks require that a UAV exhibit autonomous operation including *collision avoidance*. UAVs flying "nap of the earth" below the treetops risk collision with obstacles whose position cannot be guaranteed as known before flight. UAVs must include situational awareness based on sensing and perception of the immediate environment to locate collision dangers and plan an appropriate avoidance path [5]. Hence the desired autonomous intelligent control architecture for UAVs integrates threat/obstacle awareness with intelligent decision making, path planning and trajectory generation to achieve effective threat avoidance and mission completion. This is a complex problem that has not been successfully solved yet. If, in addition, threat/obstacle avoidance needs to be accomplished during a high-speed flight, in low visibility, cluttered environment, and under subsystem and/or component failures, the problem becomes truly formidable.

Sensors considered for collision detection include active or passive sensors. Active RADAR or LIDAR (light detecting and ranging) sensors for manned aircraft are currently under investigation for use in UAVs [6], [7]. These sensors provide resolution appropriate for wire detection, but exhibit sparse measurements, non-covert operation due to emitted radiation, and a form factor and power requirement that does not currently scale to the smallest micro air vehicles (MAVs). Passive sensors based on visual electro-optical (EO) or forward looking infrared (FLIR) are promising due to low size weight and power requirements and a lack of emitted radiation, but require significant image processing to detect obstacles. Bhanu et al. [5] argue for a *maximally passive* system that combines narrow field of view active sensors for wire detection with wide field of view passive stereo sensors for peripheral visibility. This paper proposes a passive stereo system for visual obstacle detection suitable for integration into such a maximally passive system.

In this paper, we describe the development and implementation of the Visual Threat Awareness Avoidance (VISTA) system for passive, stereo image based obstacle detection, its integration with the Multi-layer Architecture for Trajectory Replanning and Intelligent plan eXecution (MATRIX) for autonomous intelligent control of UAVs, and performance evaluation of the integrated system through flight tests. The VISTA system combines block matching stereo computed on the Acadia I vision processor designed by the Sarnoff Corporation [8] with image segmentation based on a special purpose graph representation appropriate for collision detection, human vision inspired perceptual organization and efficient graph partitioning based on the recursive minimum s-t graph cut. This segmentation provides a means to increase robustness to stereo correspondence errors as will be described in this paper, and provides constraints suitable for motion planning and avoidance. This

This work was supported under a DARPA IXO Software Enabled Control (SEC) program grant DAAH01-00-C-R187 under the direction of John Bay.

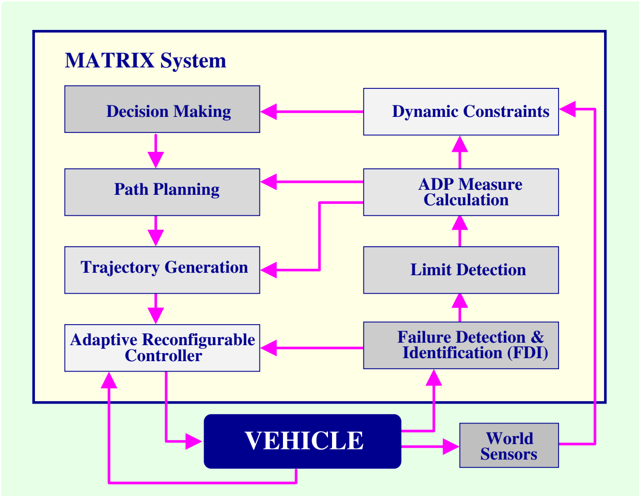


Fig. 1. Structure of the Multi-layer Architecture for Trajectory Re-planning and Intelligent plan eXecution (MATRIX) (ADP - Achievable Dynamic Performance) (©*1999-2005 Scientific Systems Company, Inc.*)

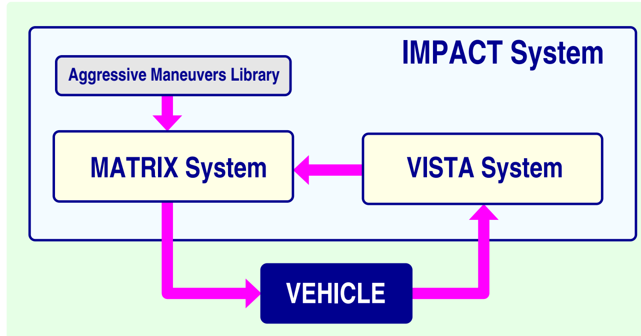


Fig. 2. Structure of the Integrated Motion Planning, Awareness and Control Technology (IMPACT) System (©*1999-2005 Scientific Systems Company, Inc.*)

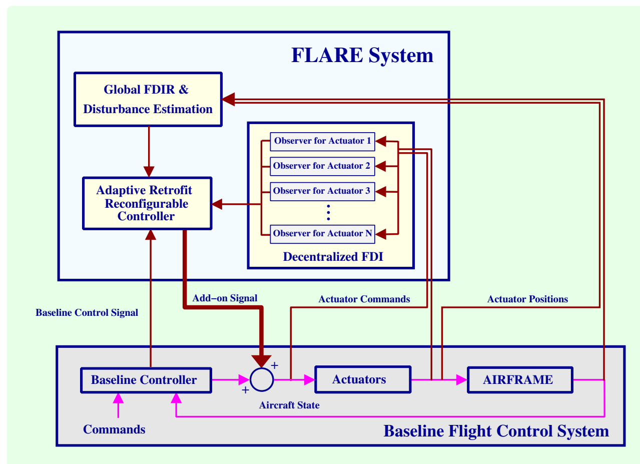


Fig. 3. Structure of the Fast on-Line Actuator Reconfiguration Enhancement (FLARE) System (©*1999-2005 Scientific Systems Company, Inc.*)

paper will describe system theory, and show experimental results from a VISTA prototype flight tested on Georgia Tech's GT-Max autonomous helicopter [9].

II. MATRIX System Architecture

We recently completed a DARPA Phase II SBIR, that was also a part of the DARPA Software Enabled Control (SEC) Program. The objectives under the project were as follows: (i) To study the issues arising in the context of autonomous control of Unmanned Aerial Vehicles and develop a multi-layer architecture for autonomous control of UAV; (ii) To implement the architecture on autonomous helicopters; and (iii) To develop and implement VISTA system (Visual Threat Awareness). Under the project we developed a new Multi-layer Architecture for Trajectory Replanning and Intelligent plan eXecution (MATRIX) system, Figure 1, and integrated it with VISTA to achieve real-time threat detection and avoidance under faults and failures. The main role of the MATRIX system is to integrate threat detection algorithms with on-line path planning and trajectory generation within an effective multi-layer architecture for pop-up threat avoidance under subsystem and component faults and failures.

An architecture that integrates the MATRIX and VISTA systems is referred to as the Integrated Motion Planning, Awareness and Control Technology (IMPACT) system, and integrates pop-up threat detection with on-line motion planning for aggressive maneuvering to achieve mission objectives for UAVs under different threats and dynamic changes in the environment. The IMPACT architecture is shown in Figure 2. Different layers of the MATRIX architecture are described next.

A. Failure Detection, Identification and Reconfiguration

The main role of this layer is to monitor the health of UAV subsystems and components, detect faults, failures and structural damage, and reconfigure the controls to achieve effective failure and damage accommodation while maintaining or gracefully degrading the desired flight performance.

We have developed several efficient algorithms for effective FDIR in the presence of actuator failures, control effector damage, and damage-generated disturbances. The main architecture that was developed is referred to as FLARE (Fast on-Line Actuator Reconfiguration Enhancement), and is shown in Figure 3. It is seen that the actuator health status is monitored by multiple decentralized FDI observers, while the damage conditions and disturbances are detected by the Global FDI system. The FDI information is passed on to the retrofit reconfigurable controller that assures fast reconfiguration and system stability. The main features of the FLARE system are as follows: (a) Fast on-line detection of failures and battle damage using low-order observers and a small number of failure-related parameters. (b) Highly robust adaptive reconfigurable control for failure and damage accommodation. (c) Capability to handle multiple failures and damages, as well as failure recoveries. (d) The reconfigurable controller is implemented in a retrofit fashion which allows the baseline flight controller to be retained. (e)

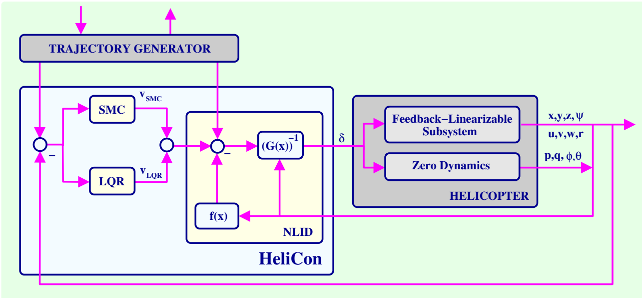


Fig. 4. Structure of the SSCI's Nonlinear Helicopter Controller (HeliCon) (*©1999-2005 Scientific Systems Company, Inc.*)

First- or second-order actuator dynamics and position and rate limits on the control effectors are explicitly taken into account in the algorithm. The FLARE system was recently evaluated through piloted F/A-18 aircraft simulations at Boeing and NASA Dryden yielding excellent results in the presence of severe flight-critical failures [2], [4].

The FLARE architecture was also implemented under the DARPA SEC program to a small autonomous helicopter. We developed a new baseline helicopter controller (HeliCon) controller that combines in an innovative way partial feedback linearization and sliding mode control with zero dynamics stabilization using outer-loop LQR controller, Figure 4. The controller was extended to add a retrofit module for accommodation of loss-of-effectiveness failure of flight control actuators, and the extended algorithm was flight tested using Georgia Tech's RMax helicopter under the DARPA SEC Program.

B. Achievable Dynamic Performance (ADP)

The main distinguishing feature of the MATRIX architecture is the Achievable Dynamics Performance (ADP) block. ADP is defined as the *maximum performance that the vehicle can achieve under different faults, failures, and external disturbances in a dynamically varying environment*. In the MATRIX architecture, an ADP measure is calculated on-line at the inner-loop control level, and passed on to the higher hierarchical levels that make appropriate changes to reflect the new lowered capabilities of the vehicle. We implemented the ADP concept under the DARPA SEC program by on-line identification of the position limits of the helicopter rudder, and ADP measure calculation based on this estimate. This measure was used by the higher hierarchical layers as described below.

C. Autonomous Trajectory Generation (ATG) Layer

The role of this layer is to fit a feasible trajectory through the way-points even while satisfying the state, control input, and spatial constraints. Trajectory generation is commonly based on minimization of a given criterion (e.g. time between the way points, fuel consumption, or low exposure to known stationary threats), and can be generated either on-line or off-line. In the case of failures, upsets, or other anticipated or unanticipated events, the path planning layer automatically reconfigures the desired path by modifying the way-points, while the trajectory generation layer fits a feasible trajectory that is achievable under the circumstances.

We have developed several trajectory generation algorithms based on splines and higher-order polynomials.

D. Autonomous Path Planning (APP) Layer

The role of this layer is to generate the motion plan for the overall mission, and compute spatial and other constraints needed for the design of the desired trajectories. Many of the routes and constraints can be computed off-line to cover different situations, including the nominal case and a set of anticipated events, and stored in memory. The constraints are computed in the form of safe set boundaries around the way-points. We have developed the path-planning algorithms based on the following techniques: (i) Voronoi diagrams and Delaunay triangulation; (ii) Mixed-integer/LMI algorithms; and (iii) Rapidly-exploring Random Trees (RRT). The latter approach is currently being implemented under a NASA Ames Phase II STTR with UC Berkeley.

E. Autonomous Decision-Making (ADM) Layer

This layer has the information about the overall mission objectives and constraints. This information, in conjunction with the sensory and ADP information and situational awareness, is used to make appropriate decisions as trade-offs between the mission success and vehicle survivability. This layer is responsible for collision avoidance, conflict resolution, mission retasking, and goal reassessment.

Under the Final Demo of the DARPA SEC Program, we demonstrated the MATRIX system through flight tests. The ADP measure was used to make a decision to retask a mission, recalculate achievable paths after a vehicle failure and fit a new feasible trajectory between the waypoints. This is shown in Figure 5.

III. VISTA System Architecture

The Visual Threat Awareness (VISTA) system is an approach to collision obstacle detection based on real time stereo, graph partitioning, perceptual organization and feature tracking. A block diagram of the system is shown in figure 6. A stereo pair of cameras is mounted forward looking on the UAV to monitor the region through which the UAV will fly. On each iteration, imagery is captured from a calibrated stereo pair of cameras and passed to the Acadia I vision processor which computes a *disparity map*. The disparity map is proportional to the scene depth, or distance to points within the scene. The imagery and disparity maps are *foveated* using a log-polar mapping compression and fused into an *affinity graph* representation using perceptual organization techniques. The affinity graph is recursively bipartitioned using a minimum s-t graph cut resulting in an estimate of k *regions* within the imagery. Statistics are computed for each region, and those that pass

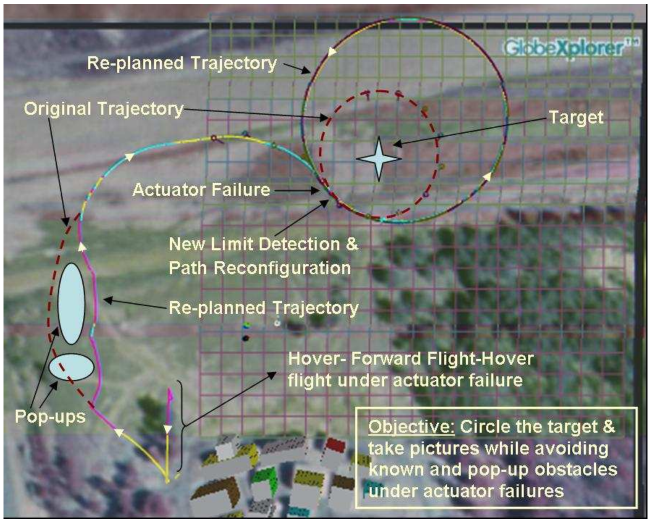


Fig. 5. Final Demo under the DARPA SEC Program: The control system first replans the trajectory to avoid a pop-up threat in the no-failure case. Following that, a rudder lock-in-place occurs that changes the position limits; this is detected by the Limit Detection System that calculates a new ADP measure and passes this information to the ADM layer that makes a decision to retask the mission (follow a larger-radius circle); this information os passed on to the APP layer that calculates new waypoints, and the ATG layer that fit a new trajectory through these waypoints.

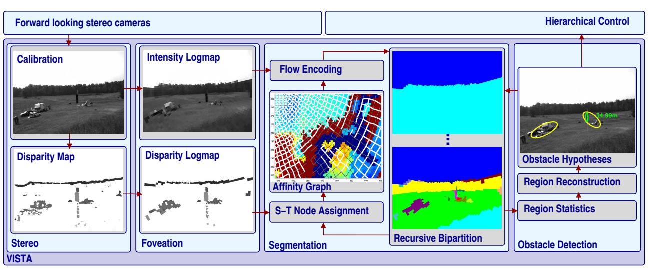


Fig. 6. Visual Threat Awareness (VISTA) system block diagram

a statistical test are reconstructed using stereo triangulation and represented with a bounding ellipse. These regions are tracked using a Kalman filter and those regions with a given tracking confidence above a threshold are labeled *obstacle hypotheses*. Obstacle hypotheses that fall within the flight path are labeled *collision obstacle hypotheses* with the closest collision obstacle labeled *nearest collision obstacle*. Collision obstacle hypotheses are measurements of the position and size of possible collision dangers which provide dynamic constraints for avoidance.

A. Computational Stereo

Computational stereo is the process of extracting three-dimensional scene structure from two or more images taken from distinct viewpoints [10]. This computation requires a three step process of calibration, correspondence and reconstruction.

Stereo calibration is the process of measuring the parameters which define the camera intrinsics, stereo intrinsics and stereo extrinsics. The camera intrinsic parameters or *camera intrinsics* define a transformation between 3D scene

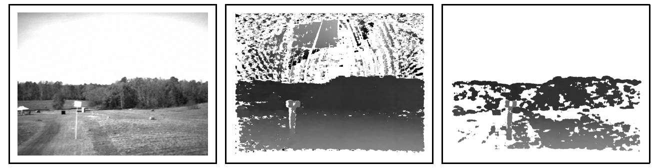


Fig. 7. Tradeoff between stereo threshold and correspondence quality. (left) Grayscale imagery (middle) Disparity map with low threshold (right) Disparity map with high threshold. Dark gray=far,light gray=close,white=undefined

coordinates and 2D image coordinates that take into account uncertainties introduced in the camera manufacturing process, geometric lens distortion and other nonlinearities. In this system, camera intrinsic calibration is a coupled process of radial lens distortion correction and camera projection matrix estimation using the approach described in [1].

Stereo correspondence is the process of establishing matching points in stereo imagery. A point at a finite distance from a stereo pair will exhibit a *disparity* or change in position between matching points in each image due to the change in viewpoint. Stereo geometry constraints the position of matching points to be along epipolar lines in the image, and calibrated stereo pairs in epipolar alignment further constrain the position to be along an image scanline. Stereo correspondence techniques attempt to find matching points in the left and right imagery by exploiting constraints such as epipolar geometry, ordering, brightness constancy, edge consistency and uniqueness [10]. However, this matching can be ambiguous when features in one image do not have an identical and unique match in the other image. This may be due to viewpoint (foreshortening), multiple feature match (regions of low contrast, periodic features) or no feature match (specular reflections, occlusion, minimum distance violation). Many correspondence techniques include a matching confidence threshold to discard poor matches, however as shown in 7, the quality of the correspondence is sensitive to this threshold. The low threshold disparity map in 7b introduces severe matching errors in the sky due to low contrast, but exhibits excellent smooth correspondence on the ground. The high threshold disparity map in 7c removes the sky errors, but also removes some correct correspondence on the ground. It is unclear how to choose this threshold in general, without introducing false alarms or missed detections in an unconstrained outdoor environment. This point will be revisited in the next section.

In this system, stereo correspondence is computed on the Acadia I vision processor using a sum of absolute differences (SAD) block matching approach along epipolar scanlines, with left/right consistency checking and maximum 32 disparity search [8]. SAD estimates are thresholded, and those points with SAD measure above this threshold define a *disparity map* which is proportional to scene depth using stereo reconstruction. The Acadia I vision processor is dedicated to stereo processing, resulting in 640x480 disparity map computation at 23Hz. Other similar

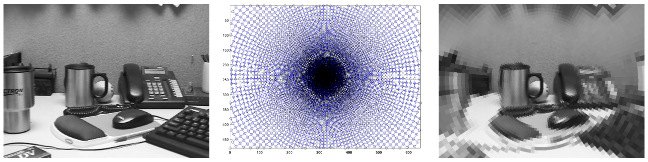


Fig. 8. Example of foveation using the log-polar mapping

commercial approaches that achieve similar performance using variants of the block matching technique include Videre Design, Point Grey, Tyzx.

Finally, stereo reconstruction is the computation of depth from disparity determined from correspondence and stereo geometry determined from calibration This reconstruction uses standard stereo triangulation to recover 3D scene structure from 2D projections [11], resulting in depth measurements to points in the scene, which provides collision distance for obstacle detection.

B. Foveation

Foveation refers to a space variant image representation with a high resolution central region or *fovea* surrounded by a lower resolution periphery [12], [13]. In the context of collision detection, foveation provides the benefits of compression and focus of attention. Collision detection systems exhibit a tradeoff between sensor resolution for detection of small obstacles, and detection time requirements for safe operation. Computational complexity is proportional to sensor resolution, so limited computational resources require that computation is focused appropriately. Foveation retains high resolution in the image center, which has a high likelihood of containing a collision obstacle since the image center of a forward looking sensor can be actively aligned with the current heading. Foveation also reduces the resolution in the periphery which may contain an obstacle, but a low likelihood of containing a collision obstacle. Therefore, foveation allows the system to focus available computational resources on those spatial image regions that are likely to contain collision dangers. The lower resolution periphery provides image compression that is appropriate for collision detection, and focuses computation on the image center which is likely to contain collision dangers.

Foveation can be implemented using a *log-polar mapping* [13], such that the space variant resolution is proportional to the log of the distance from the image center. An example of the log-polar mapping is shown in figure 8. Pixels in figure 8a are mapped to nearest log-polar sectors with centroids represented as circles in figure 8b, such that the median grayscale intensity represents the entire sector in the log-polar mapping as in figure 8c.

C. Segmentation

Segmentation can be defined as the process of *labeling* an image such that features with equal labels are "similar" and features with unequal labels are "dissimilar". A labeling defines groupings of pixels into *regions* such that pixels with a common label belong together in some sense, and pixels with different labels do not. In the context of obstacle detection, segmentation provides hypothesized obstacle size and obstacle boundaries. Segmentation groups pixels into regions which are used to define the extent of an obstacle hypothesis for motion planning. Hypothesized obstacle boundaries are boundaries between segmentation labels, which can be used to compensate for the stereo correspondence errors described in section III-A by ignoring disparity within groups and enforcing the edge consistency constraint along a label boundary. This will be revisited in the next section.

The segmentation problem can be posed formally as an energy minimization problem [14]. Assume that there exists a finite set of points $P = \{p_1, p_2, \dots, p_N\}$ that fall within the field of view of the sensor for which measurements can be taken. For each point p_i, a sensor can capture a multidimensional measurement $M(p_i) = \{m_1, m_2, \dots, m_k\}$, such that the total set of all measurements for all points is $S = \{M(p_1), M(p_2), \dots, M(p_N)\}$. Each measurement is some descriptive feature of p_i that may include intensity, texture, color, intensity gradient, motion, depth or others. A *labeling* $f(P)$ is a mapping from P to L where L is a finite set of labels. An *energy optimal labeling* f^* minimizes the energy function E [15]

$$E(f) = \sum_{p \in P} D_p(f_p) + \sum_{(p,q) \in N} V_{p,q}(f_p, f_q) \tag{1}$$

$$f^* = \arg\min_f E(f) \tag{2}$$

D_p is a function which encodes the cost of assigning label f_p to p, which represents prior knowledge about the true labeling of p. $V_{p,q}$ is a function which encodes the cost of assigning label f_p to p and a different label f_q to q when (p, q) are *neighbors* in a given neighborhood set $N \subset P \times P$. This function represents a penalty for violating label smoothness for neighboring (p, q). Solutions f^* to the energy minimization problem are difficult to find in general since (1) can be non-convex in a high dimensional space.

In this application, we approach the energy minimization in (2) as a *recursive maximum network flow* problem. A network flow graph is defined as a directed graph $G = (V, E, W)$ with nodes V, edges E and edge weights W. Edge weights w_{ij} between nodes i and j are interpreted as *capacities*, and certain distinguished nodes s and t in V are interpreted as *terminal nodes*. The maximum network flow problem is that of determining the maximum flow of some commodity between terminal nodes such that the maximum flow on any edge is less than or equal to capacity [16]. Using the Ford-Fulkerson theorem, it can be shown that a solution to the maximum network flow or *maxflow* problem is also a solution to the minimum graph cut or *mincut* problem [17]. The mincut on a network flow graph defines a graph bipartition which is equivalent to a *binary* labeling. This binary labeling is an exact solution to the energy minimization in (2) assuming that D_p is equal to the terminal edge capacities and $V_{p,q}$ is equal to the edge capacities such that $V_{p,q}$ is a *regular function* as defined in [15]. Recursive application of the binary labeling generates a k-labeling such that the maximum of the inter-partition flows is minimizes among all possible partitions of G into the same number of partitions [18].

This approach requires that imagery be abstracted to a network flow graph representation suitable for obstacle detection. Graph nodes and graph edges are defined using the log polar mapping from section III-B. Graph edge weights are encoded using *perceptual organization* heuristic of similarity such that the edge weight w_{ij} or *node affinity* between adjacent nodes i and j can be defined as:

$$w_{ij} = \sum_{d=1}^{D} \alpha_d d_{ij} \quad (3)$$

$$d_{ij} = \exp\left(\frac{-(\max((m_d(i) - m_d(j)), \mu_d) - \mu_d)^2}{2\sigma_d^2}\right) \quad (4)$$

Equation 4 is a nonlinear model for *feature smoothness*, which is similar to the approach by Shi and Malik in [19]. Each node i has D measurements $m(i) = [m_1 m_2 \ldots m_D]^T$, with the measurement $m_d(i)$ corresponding to the d^{th} measurement for the i^{th} node, such as intensity, color, texture, depth or others. The feature smoothness between nodes i and j is parameterized by μ_d and σ_d which define the mean and standard deviation for smooth feature changes. The resulting node affinity is a linear combination of each feature smoothness with weight α_d. For this application, features include median intensity and median disparity. Graph terminal nodes encode the hypothesis that certain nodes represent foreground and background. Therefore, those nodes that exhibit depth contrast, and are near to the sensor are likely to be foreground and those further from the sensor are likely to be background [1]. Finally, the minimum cut of the network flow graph is computed using a new polynomial time augmenting path approach proposed by Boykov and Kolmogorov [20]. Additional issues of oversegmentation, perceptual organization and Rubin's rules, terminal node assignment, recursion stopping criterion, coefficient α_d learning and single node groups are addressed in [1].

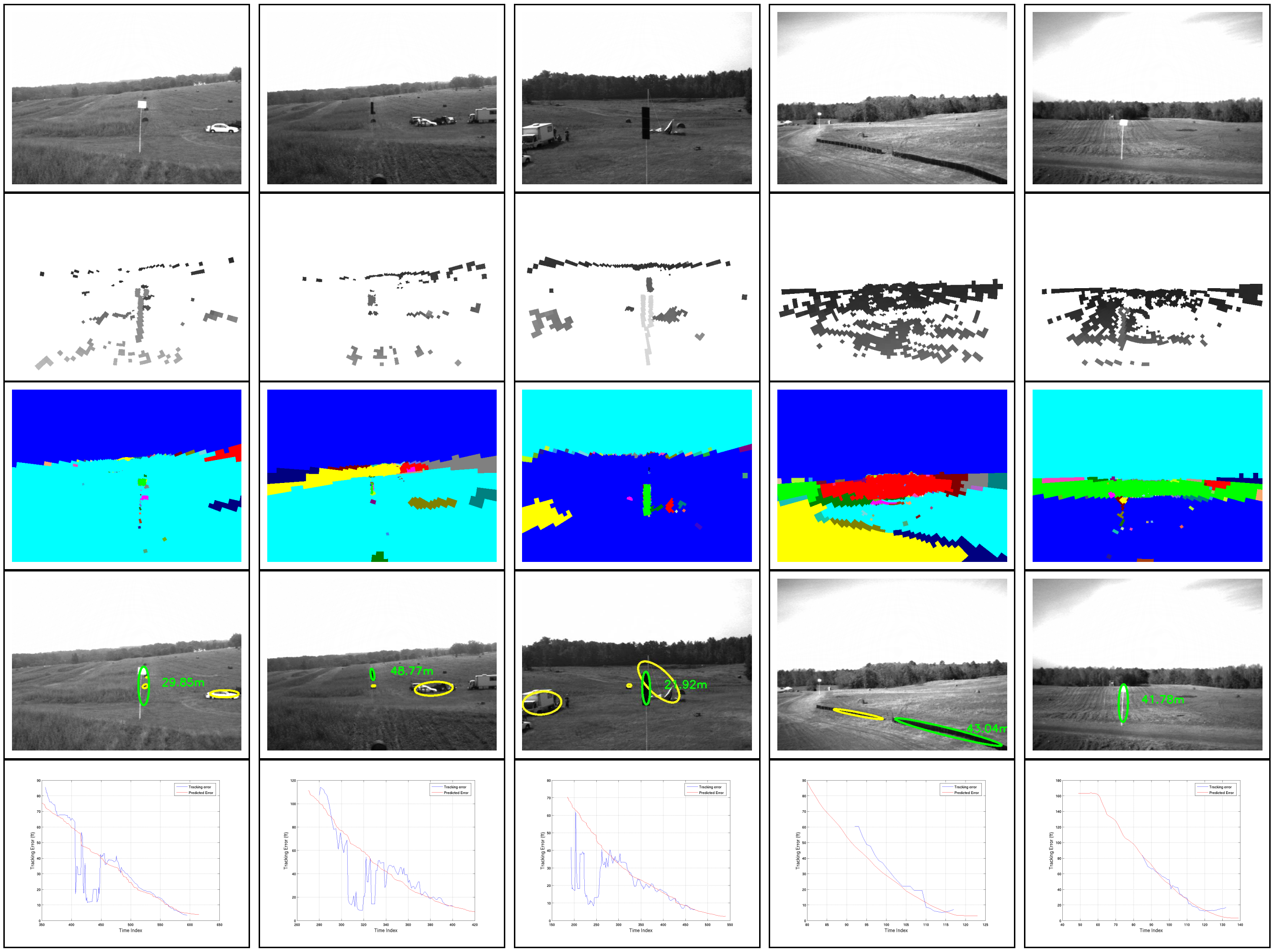
Example segmentations are shown in figure 9 (row three), where regions of constant color have the same label. An example of the first and last steps in the recursive bipartition are shown in figure 6.

D. Obstacle Detection and Tracking

Obstacle detection and tracking includes boundary statistics, region reconstruction and obstacle tracking. The k-partition from section III-C defines a set of k-regions in the image which must be reconstructed in 3D using the stereo geometry and disparity. For each region, we compute *boundary statistics* for measurements about the boundary of each region. Those regions with statistics above a given threshold are reconstructed in 3D using the boundary disparity. Region centroids are used to determine if the region falls within the tracking volume. Such regions are parameterized by the bounding ellipse, and tracked using a Kalman filter.

Boundary statistics can be used to compensate for stereo correspondence errors by ignoring disparity in region interiors and by checking the *edge consistency constraint* along region boundaries. Boundary statistics are those statistics which are computed over all feature measurements at the boundary of a given region. As discussed in section III-A, noisy stereo disparity estimates may be introduced due to poorly chosen stereo threshold or stereo correspondence errors from scene geometry. Noisy disparity results in incorrect 3D reconstruction which can generate false alarm obstacles, or missed obstacles altogether. Stereo correspondence is strongest in areas exhibiting intensity edges corresponding to local maxima in intensity gradients. By nature of the segmentation process and the formulation of node affinity, the interior of a segmented region will exhibit smooth changes in feature measurements, and the boundary will exhibit violations of smoothness. Therefore, he boundary of a segment will exhibit stronger correspondence than the interior, which means the disparity interior to a region can be discarded in favor of the disparity at the boundary. In other words, disparity from regions of low contrast is ignored. The *edge consistency* constraint is commonly used in computational stereo to constrain the search for correspondence [21], such that disparity along intensity edges should be smoothly varying. Any violation of the edge consistency constraint is an indication of incorrect correspondence. Therefore, we define an edge consistency check in terms of disparity variance along a boundary, such that a region with a boundary variance above a threshold violates edge consistency and is discarded.

Those regions which pass the edge consistency check are reconstructed in 3D using the bounding ellipse of the region. Bounding ellipses which fall within a given tracking volume are labeled *obstacle hypotheses* and bounding ellipses which fall within a given collision volume are labeled *collision obstacle hypotheses*. Those ellipses outside the tracking volume are ignored for computational efficiently, The ellipse parameters for obstacle hypotheses are then passed as measurements to a Kalman Filter for obstacle tracking [22]. Each obstacle is tracked independently such that obstacles which enter and exit the tracking volume spawn or destroy their associated filter. Measurement assignment is determined by comparing the measurement to all obstacles within a specified gating distance. Measurements are assigned to the obstacle with the minimum error in ellipse parameters and closest mean intensity. The result is a state estimate $\hat{x}_k$ and state estimate covariance P_k for the bounding ellipse of each obstacle in the inertial frame of the vehicle. Detected obstacles within the collision volume are then passed to the control system for motion planning.



Flight 1 Flight 2 Flight 3 Flight 4 Flight 5

Fig. 9. VISTA sample imagery and flight experiment results for collision detection scenarios. From top to bottom: calibrated grayscale imagery from left camera, foveated disparity map (dark gray-far, light gray=close, white=undefined), k-partition segmentation (solid color=region), obstacle detection (yellow ellipse=tracked obstacle, green ellipse=nearest collision obstacle, green text=collision distance to nearest collision obstacle), obstacle detection performance evaluation

IV. Flight Experiments

Flight experiments for the VISTA system were performed on the Georgia Tech GT-Max autonomous helicopter platform [9], outfitted with the VISTA flight computer and stereo cameras. Flights 1,4,5 had the helicopter autonomously approach a ”sign” obstacle, which was a 40”x30” piece of white foamcore mounted on the top of a 21’ tall, 0.75” diameter pole. Flights 2,3 replaced the ”sign” with a ”pole” obstacle, which was a 90”x20” piece of black foamcore representing the top section of a 20” diameter telephone pole. The helicopter approached the obstacles at a constant velocity and altitude, with variable heading (north/south or east/west), forward speed and ambient lighting for each flight. Flight experiments were performed in a field in McDonough GA that included hay bales, trees, tarpaulins, gantry and ground station vehicles in the background.

Figure 9 shows sample imagery and processing results from five flight experiments. Flight data includes calibrated grayscale imagery, disparity maps, segmentation results and obstacle detection. The obstacle detection imagery shows that the nearest collision obstacle is detected as shown with a green ellipse, but also additional obstacles are detected as shown with yellow ellipses. These obstacles include cars, hay bales, tarpaulins and a gantry in the background that are in fact obstacles which are corrected detected by the system.

Figure 9 also show a graph of obstacle detection performance. The ground truth position of the collision obstacle was captured after each flight, and the obstacle estimation error was computed by comparing the tracking estimate of the green ellipse centroid to the ground truth obstacle centroid. The detection error graphs show the Euclidean distance between the estimated position of the obstacle $\hat{P}$ and the ground truth position P, such that the error at time index i $E_i = |P - \hat{P}|$. The position estimation error is shown in blue. The red plot shows the predicted estimation error given the ground truth distance to the obstacle and the known resolution of stereo. Stereo estimation error reflects the nonlinear range resolution of stereo due to pixel quantization, such that the uncertainty in range for a single disparity is proportional to the square of the range. At time index i, the helicopter with position P_h is at range $r_h = |P - P_h|$ from the obstacle. The stereo range resolution at distance r_h is given by

$$\Delta_z(r_h) = \frac{r_h^2}{Bf} \tag{5}$$

for a known baseline B in meters and focal length f in pixel units. This range resolution is the expected uncertainty for a single disparity which is due to pixel quantization error and stereo geometry. A correct detection result should track the error $E(r_h) \approx 2\Delta_z(r_h)$, where E is the range uncertainty due to a two pixel disparity error. The error E is also a function of the vertical and horizontal position error, but these errors is dominated by range uncertainty for the distances considered in UAV flight, and are negligible in practice [23]. The plots show that the position estimation error does track E, and at times improves on the expected error due to subpixel disparity estimates from tracking and from disparity averaging.

The obstacle detection performance results include processing results for the entire run, including the period in which the helicopter is pitching down during acceleration, and pitching up during halt. The plots begin at the first time index in which the obstacle is detected, which shows that there are no false alarms. Runtime performance for each flight ranged from 5Hz-10Hz, with variations due to scene complexity affecting the total number of regions k of the recursive bipartition. Future work includes extending the analysis to multi-obstacle scenarios, additional trajectories and more complex urban environments.

REFERENCES

[1] Scientific System Co. Inc, “Development of Intelligent Reconfigurable Control Algorithms and a Software Design Toolbox for Autonomous Systems”, DARPA Phase II SBIR Final Report, Contract No. DAAH01-00-C-R187, March 2005.

[2] J. D. Bošković, S. E. Bergstrom, R. K. Mehra, James Urnes, Sr., Mark Hood, and Yohan Lin, ”Fast on-Line Actuator Reconfiguration Enabling (FLARE) System”, to be presented at the *2005 AIAA Guidance, Navigation and Control Conference*, San Francisco, CA, August 15-18, 2005.

[3] J. D. Bošković, R. Prasanth and R. K. Mehra, ”A Multi-Layer Autonomous Intelligent Control Architecture for Unmanned Aerial Vehicles”, *AIAA Journal of Aerospace Computing, Information, and Communication (JACIC)*, Vol. 1, pp. 605-628, December 2004.

[4] J. D. Bošković and R. K. Mehra, ”Robust Integrated Flight Control Design Under Failures, Damage and State-Dependent Disturbances”, *AIAA Journal of Guidance, Control & Dynamics* (in press).

[5] B. Bhanu, B. Roberts, D. Duncan, and S. Das, “A system for obstacle detection during rotorcraft low-altitude flight,” *IEEE Transactions on Aerospace and Electronic Systems*, vol. 32, no. 3, pp. 785–897, July 1996.

[6] Russell C. Wolfe, “Nasa erast non-cooperative dsa flight test,” in *Proceedings of AUVSI Conference*, Baltimore, July 2003.

[7] M. Whalley, M. Freed, M. Takahashi, D. Christian, A. Patterson-Hine, G. Schulein, and R. Harris, “The nasa/army autonomous rotorcraft project,” in *59th American Helicopter Society Annual Forum*, Phoenix, AZ, May 2003.

[8] G. van der Wal, M. Hansen, and M. Piacentino, “The acadia vision processor,” in *IEEE proceedings of International Workshop on Computer Architecture for Machine Perception (CAMPS 2000)*, September 2000.

[9] E. Johnson and D. Schrage, “The georgia tech unmanned aerial research vehicle: Gtmax,” in *Proceedings of the AIAA Guidance, Navigation, and Control Conference*, 2003.

[10] M. Brown, Z. Burschka, and G.D. Hager, “Advances in computational stereo,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 25, no. 9, pp. 993–1008, 2003.

[11] O. Faugeras, *Three-Dimensional Computer Vision: a Geometric Viewpoint*, MIT press, 1993.

[12] E.L. Schwartz, D.N. Greve, and G. Bonmassar, “Space-variant active vision: Definition, overview and examples,” *Neural Networks*, vol. 8, no. 7-8, pp. 1297–1308, 1995.

[13] Richard Wallace, Ping-Wen Ong, Ben Bederson, and Eric Schwartz, “Space variant image processing,” *International Journal of Computer Vision*, vol. 13, no. 1, pp. 71–90, 1994.

[14] B.T. Porteous D. Greig and A. Seheult, “Exact maximum a posteriori estimation for binary images,” *Journal of the Royal Statistical Society*, vol. 51, no. 2, pp. 271–279, 1989.

[15] Vladimir Kolmogorov and Ramin Zabih, “What energy functions can be minimized via graph cuts?,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 26, no. 2, pp. 147–159, February 2004.

[16] R. K. Ahuja, T. L. Magnanti, and J. B. Orlin, *Network Flows: Theory, Algorithms, and Applications*, Prentice Hall, 1993.

[17] Sr. L.R. Ford and E. Fulkerson, *Flows in Networks*, Princeton University Press, 1962.

[18] Z. Wu and R. Leahy, “An optimal graph theoretic approach to data clustering: Theory and application to image segmentation,” *IEEE trans. on Pattern Analysis and Machine Intelligence*, vol. 15, pp. 1101–1113, November 1993.

[19] J.Shi and J.Malik, “Normalized cuts and image segmentation,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 22, no. 8, pp. 888–905, August 2000.

[20] Yuri Boykov and Vladimir Kolmogorov, “An experimental comparison of min-cut/max-flow algorithms for energy minimization in computer vision,” in *International Workshop on Energy Minimization Methods in Computer Vision and Pattern Recognition*. September 2001, vol. 2134 of *LNCS*, pp. 359–374, Springer-Verlag.

[21] Y. Ohta and T. Kanade, “Stereo by intra- and inter-scanline search using dynamic programming,” *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 7, no. 1, pp. 139–154, 1985.

[22] A. Gelb, Ed., *Applied Optimal Estimation*, MIT Press, 1974.

[23] S.D. Blostein and T.S. Huang, “Error analysis in stereo determination of 3d point positions,” *IEEE Transactions Pattern Analysis and Machine Intelligence*, vol. 9, no. 6, pp. 752–765, November 1987.