Conflict-Free Navigation in Unknown Urban Environments

Autonomous Exploration for Unmanned Aerial Vehicles

BY DAVID H. SHIM, HOAM CHUNG, AND S. SHANKAR SASTRY

Recently, there has been a great interest in the development of advanced unmanned aerial vehicles (UAVs) capable of missions in complex dynamic environments. Conventional waypoint-based navigation systems are typically unable to sense and avoid obstacles, and, therefore, they are not suitable for missions in cluttered urban environments. Advanced flight control systems for urban navigation should be able to adjust the flight path dynamically with the information on the surroundings collected using onboard sensors in real time.

Autonomous exploration in an unknown environment requires a map generation on the surroundings and path planning for collision-free navigation. These topics have been intensively covered by the robotics community since the late 1970s. Based on these efforts, various algorithms and implementations are currently available for guiding mobile robots in an unknown or partially known two-dimensional (2-D) world. In the 1990s, researchers introduced probability theories into map building techniques [1], enhancing robustness and performance of those algorithms even with less costly and inaccurate sensors [2]. Although some of these algorithms can be extended to three-dimensional (3-D) problems, their computational load is prohibitively large and/or they do not scale up well to problems in 3-D space. From a practical perspective, UAVs, due to their faster speed, tend to pose more challenges than the ground robots or unmanned underwater vehicles (UUVs) [7] in terms of speed and accuracy in obstacle sensing and evasion. In addition to payload limitation, during the development stage, typical trial-and-error approaches are not usually possible for UAVs because failures to avoid obstacles can lead to costly and dangerous outcomes.

Model predictive control (MPC) has been found effective to solve control problems on dynamic systems with input and state constraints [9], [10] in an explicit manner. The online optimization [3] over a finite horizon allows for a control system more perceptive to future variations in the system dynamics and operating environment. In [10], it is proposed to solve for a plausible trajectory using mixed-integer linear programming (MILP) with constraints such as obstacle avoidance. In [8], obstacle-free trajectories in urban environments are computed using a nonlinear MPC technique. However, their algorithm requires a priori information about environments, and the collision-free air space is explicitly represented by convex cones between known urban structures. Alternatively, in [4], it is shown that the MPC algorithm can be formulated to solve the stabilization and tracking problem of a nonlinear kinodynamic equation with control input saturation, state constraints, and some behavioral constraints such as collision avoidance and pursuit-evasion games. In this article, for numerical tractability and improved reliability, we propose a hierarchical flight control system that enables conflict-free navigation by tracking the trajectories generated by the real-time MPC optimization module for obstacle avoidance.

Conflict-Free Navigation on the Fly

In this article, we propose an autonomous exploration algorithm suitable for, but not limited to, urban navigation by combining the MPC-based obstacle avoidance algorithm with local obstacle map building using an onboard laser scanner. The MPC layer solves for a collision-free trajectory by a real-time gradient-search-based optimization. The tracking control

BACKGROUND IMAGE: © PHOTODISC

1070-9932/06/$20.00©2006 IEEE

Figure 1. A Berkeley UAV flying autonomously in a mock-up urban environment.

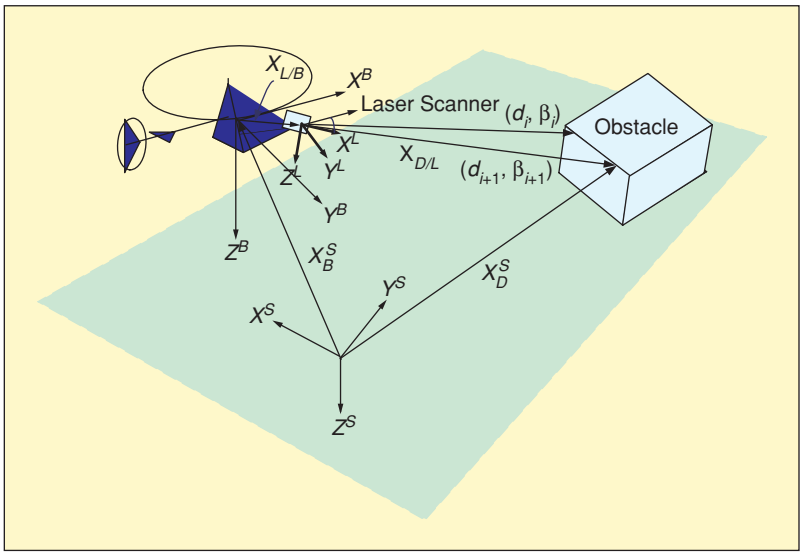

Figure 2. Coordinate transformations for laser scan data.

layer [6] is responsible for following through the given trajectory. The proposed framework is validated in simulations and then successfully tested in a series of experiments using a mock-up urban environment as shown in Figure 1.

Formulation

In this section, we present the coordinate systems definitions, coordinate transformation, and the mathematic formulation of an obstacle avoidance problem in MPC framework.

Definitions of Coordinate Systems and Transformations

The laser scanner used in this research consists of a laser source, a photoreceptor, and a rotating mirror for planar scanning. An accurate timing device measures the time lapse from the moment the laser beam is emitted to the moment the laser beam reflected on an object returns to the receptor. A rotating mirror reflects the laser beam in a circular plane, allowing for 2-D scanning.

At each scan, the sensor reports a set of measurements that supplies the following measurement set:

$$Y_L = \{(d_n, \beta_n), n = 1, \ldots, N_{\text{meas}}\}, \tag{1}$$

where d_n, β_n, and N_{meas} represent the distance from an object, the angle in the scanning plane, and the total number of measurements per scan, respectively. Each measurement, i.e., the relative distance from the laser scanner to a scanned point in the laser-scanner coordinate system, can be written into a vector form such that

$$\mathbf{X}_{D/L}^L|_n = d_n(\cos\beta_n \mathbf{i}^L + \sin\beta_n \mathbf{j}^L), \tag{2}$$

where $\mathbf{i}^L$ and $\mathbf{j}^L$ are orthonormal unit vectors in X^L and Y^L directions on the scanning plane, respectively. D, L, B, and S represent the scanned data, laser scanner, vehicle body coordinate system, and spatial coordinate system, respectively (Figure 2).

The calculation of the spatial coordinates of detected points involves a series of coordinate transformations among three coordinate systems: body coordinate systems attached to the laser scanner and to the host vehicle and the spatial coordinate system, to which the vehicle location and attitude are referred.

Each measurement vector in the laser-scanner-attached coordinates is first transformed into the vehicle body coordinates and then the spatial coordinate system as following:

$$\begin{aligned}\mathbf{X}_{D/L}^S &= \mathbf{R}^{S/L}\mathbf{X}_{D/L}^L \\ &= \mathbf{R}^{S/B}\mathbf{R}^{B/L}(\alpha)\mathbf{X}_{D/L}^L.\end{aligned} \tag{3}$$

$\mathbf{R}^{B/L}(\alpha)$ is the transformation matrix from the laser body coordinate L to vehicle body coordinate B, where α is the tilt angle with respect to the vehicle body coordinate system. $\mathbf{R}^{S/B}$ denotes the transformation matrix from vehicle body coordinates to spatial coordinates.

Finally, the spatial coordinate of the obstacle is found by

$$\begin{aligned}\mathbf{X}_D^S &= \mathbf{X}_{D/L}^S + \mathbf{X}_{L/B}^S + \mathbf{X}_B^S \\ &= \mathbf{R}^{S/B}\mathbf{R}^{B/L}(\alpha)\mathbf{X}_{D/L}^L + \mathbf{R}^{S/B}\mathbf{X}_{L/B}^B + \mathbf{X}_B^S.\end{aligned} \tag{4}$$

Using (4), one can find the spatial coordinate of sampled points on obstacles by combining the raw measurement vector with the position, heading, and attitude of the vehicle, which are available from the onboard navigation system of the UAV. It should be noted that the detection accuracy in the spatial coordinate system not only depends on the laser scanner's accuracy itself but also on the accuracy of the vehicle states.

To ensure conflict-free navigation in an airspace filled with obstacles, the laser scanner should scan the surroundings wide enough to find conflict-free trajectories. For example, if the laser scanner is installed to scan the area horizontally, an actuation in the pitch axis is necessary so that the scanner can cover the frontal area sufficiently higher than the rotor disc plane and lower than the landing gear. Figure 3 shows actuated laser scanner mountings on Berkeley UAVs. The scanner is mounted on a tilt actuator with an encoder, which provides the tilt angle α in $\mathbf{R}^{B/L}(\alpha)$.

System Dynamics and Trajectory Generation Using MPC

A mathematic model for a given dynamic system can be written as a discrete-time dynamic equation such that [4]

$$\mathbf{x}(k+1) = \mathbf{f}(\mathbf{x}(k)) + \mathbf{g}(\mathbf{x}(k))\mathbf{u}(k). \tag{5}$$

We are interested in solving a discrete-time optimal control problem for the system in (5) to find the control input sequence such that

$$\{\mathbf{u}^*(k)\}_{k=1}^{T} = \arg\min \sum_{k=1}^{T} q(\mathbf{x}(\mathbf{k}), \mathbf{u}(\mathbf{k})) + q_f(\mathbf{u}(T+1)). \tag{6}$$

subject to the dynamic equation (5). In (6), $q(\mathbf{x}, \mathbf{u})$ is the cost function over the finite horizon, and $q_f(\mathbf{x})$ is the terminal cost.

The *nonlinear model predictive control* (NMPC) problem solves for the optimal control sequence $\{\mathbf{u}^*(k)\}_{k=1}^{T}$ with the nonlinear dynamic equation in (5) and implements the optimal input $\{\mathbf{u}^*(k)\}_{k=1}^{\tau}$ for $1 \le \tau \le T$ and then repeats these steps from the state $\mathbf{x}(\tau+1)$ at $k = \tau + 1$.

The path planning strategy in this article combines, as originally proposed in [4], the concept of a potential field with the NMPC-based online optimization. The cost function in (6) includes the potential function terms responsible for path planning in the presence of stationary or moving obstacles. This allows the trajectory generation and vehicle stabilization to be combined into a single optimization problem, and the look-ahead nature of the MPC framework makes this approach less vulnerable to local minima [4].

For obstacle avoidance, the following potential function term is included in the cost function $q(\mathbf{x}(k), \mathbf{u}(k))$

$$q^{\text{obst}}(\mathbf{x}(k)) = \sum_{i=1}^{N} \frac{K_i}{a_i(x^s(k) - x_i(k))^2 + b_i(y^s(k) - y,(k))^2 + c_i(z^s(k) - z,(k))^2 + \varepsilon}, \tag{7}$$

where (x^s, y^s, z^s) denotes the position of the vehicle, and (x_i, y_i, z_i) denotes the position of the ith obstacle in the spatial coordinate system. Equation (7) introduces a potential field near N obstacle points into the MPC framework: (a_i, b_i, c_i) is a set of scaling factors in X^S, Y^S, and Z^S directions, respectively, and ϵ is a positive constant to prevent (7) from being singular when $(x^s, y^s, z^s) = (x_i, y_i, z_i)$. In the following, we will present an obstacle avoidance algorithm using the MPC framework shown above for urban exploration problems.

Autonomous Exploration Using MPC with a Local Map

In this section, we present MPC-based trajectory generation for autonomous exploration in unknown environments with obstacles. Particularly, we are interested in addressing the safe navigation of UAVs in urban environments with no prior information available on the obstacles. We begin with the following statement:

Problem Statement

> *Find a trajectory that allows the vehicle to navigate from the given starting Point A to the destination Point B with a safe distance from obstacles in the environment.*

We address the problem with an integral approach of an MPC-based trajectory planner with local obstacle map generation using onboard sensors.

Trajectory Replanning Using MPC

The MPC-based trajectory replanning begins with a reference trajectory from Point A to Point B, given by a higher-level strategic layer. Without loss of generality, the initial reference trajectory is assumed to be a straight line. The MPC approach presented above can be formulated as a tracking control problem with a cost function term in (6) such that

$$q^{\text{trk}}(\mathbf{x}(k)) = \frac{1}{2}(\mathbf{y}_{ref}(k) - \mathbf{x}(k))^T Q(\mathbf{y}_{ref}(k) - \mathbf{x}(k)), \tag{8}$$

where $q(\mathbf{x}(k)) = q^{\text{trk}}(\mathbf{x}(k)) + q^{\text{obst}}(\mathbf{x}(k))$.

In [4], for selecting q^{obst}, it was shown that the cost term (7) with $N = 1$ is sufficient to find the solution, although (7) with $N > 1$ is expected to result in an optimization surface more favorable to the gradient-search-based optimization. In order to lower the computation load, however, we leave $N = 1$ so that

Figure 3. A laser scanning device mounted on an actuated mounting platform.

only the nearest point is considered in the optimization (Figure 4). It is noted, however, that the "nearest point" is not fixed at one point, as the state variables, including the vehicle position, evolve over the hypothetic finite horizons during the optimization. We will present practical issues for finding the nearest point using the sensor-based local map in the following.

Setting $N = 1$ and letting the cost function decay uniformly in all directions, the cost term in (7) for urban navigation is set to

$$\begin{aligned} q^{\text{obst}}(\mathbf{x}(k)) =& K_{\text{obs}}((x^s(k) - x_{\min}(k))^2 \\ &+ (y^s(k) - y_{\min}(k))^2 \\ &+ (z^s(k) - z_{\min}(k))^2 + \varepsilon)^{-1}. \end{aligned} \tag{9}$$

K_{obs} is a tuning parameter to balance the tendency to stay on the original given path and to break away from the given path to go around obstacles.

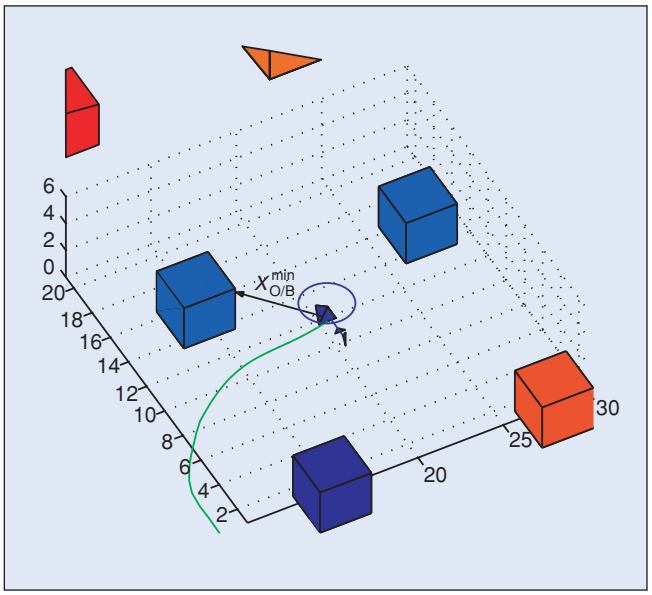

Figure 4. Nearest-point method.

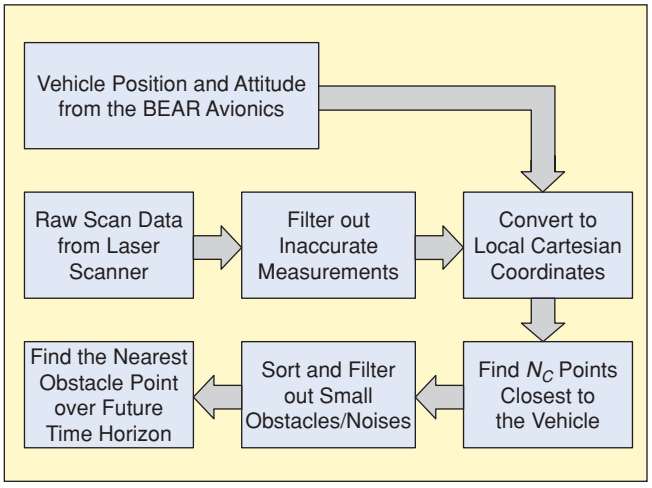

Figure 5. Local map building method for the nearest-point approach.

In [4], (5) is chosen to be the full vehicle dynamic model so that the stabilizing control input $\mathbf{u}(k)$ also minimizes other penalties for tracking, obstacle avoidance, or aerial pursuit-evasion games. In this article, to keep the computation load lower, we choose a simplified dynamic model as shown in (10) for (5):

$$\mathbf{x}(k+1) = \mathbf{x}(k) + T_s\mathbf{u}(k), \tag{10}$$

where $\mathbf{x} \triangleq [x^s\, y^s\, z^s]^T$, $\mathbf{u} \triangleq [v_x^s v_y^s v_z^s]^T$, and T_s is the sampling time of the discretized model. In this setup, the optimization result is the reference velocity in the spatial coordinates. The final trajectory is obtained by solving the forward difference (10) with the reference velocity. The vehicle trajectory of the simplified model driven by the optimal control sequence is sent to the tracking control layer [6]. By separating the trajectory layer from the stabilization layer, any failure of the optimization routine to converge to a solution can be beneficially isolated from the overall stability of the vehicle. However, due to the tracking error between the reference position and the physical position, an algorithm is conceived to find the nearest point during the optimization process based on the collected obstacle points, as discussed in the following section.

Local Obstacle Map Building

For the MPC-based trajectory generation with obstacle avoidance, we need to find $\mathbf{X}_O^{\min}$, the vector from the *reference position* to the nearest point on an obstacle such that

$$\mathbf{X}_O^{\min}(\mathbf{X}_{ref}) = \arg\min_{\mathbf{X}_O \in S_{obs}} \|\mathbf{X}_O - \mathbf{X}_{ref}\|_2, \tag{11}$$

where $\|\cdot\|_2$ is Euclidian norm in 3-D space, and S_{obs} is the set of all points on the obstacles in the surrounding 3-D space.

Theoretically, (11) demands a perfect knowledge of all obstacles in the surrounding environment, which assumes an ideal sensor capable of omnidirectional scanning with infinite detection range. Also, during the optimization, a hypothetical sensor should be moving along the trajectory of the state propagation over a finite horizon at each iteration step. Obviously, a real-world sensor would not satisfy such requirements. Finally, if the MPC algorithm is used as a reference trajectory generator, due to the inevitable tracking error, the range data is measured at the physical location of the vehicle and not on the reference trajectory. Therefore, in order to provide $\mathbf{X}_O^{\min}$ to the MPC-based trajectory generator during the optimization, it is mandatory to maintain a local obstacle map caching recent measurements from onboard sensors.

At each scan, the sensor provides N_{meas} measurements of the scan points from the nearby obstacles. Due to the imperfect coverage of the surroundings with possible measurement errors, each measurement set $\mathbf{X}_O^i$ is first filtered, transformed into local Cartesian coordinates, and cached in the local obstacle map. A first-in first-out (FIFO) buffer is chosen as the data structure for the local map, whose buffer size is determined by the types of obstacles nearby. If the surrounding is known to be static, the buffer size can be as large as the memory and

processing overheads permit. On the other hand, a more dynamic environment would require a smaller buffer to reduce the chance of detecting obstacles that may not exist any more.

In order to solve (11), the measurement set in the FIFO is sorted in ascending order of $\|\mathbf{X}_O^i - \mathbf{X}_{\text{ref}}\|_2$ for all $\mathbf{X}_O^i$ in the local obstacle map, where $1 \le i \le N_C$. Before being registered in the database, any anomalies such as "salt-and-pepper" noise should be discarded. Also, the measurements are examined for any small debris such as grass blades or leaves blown by the downwash of the rotor. Such small-size objects, not being serious threats for safety, are ignored. In order to eliminate these anomalies, we first discard measurements out of minimum and maximum detection range. Then we apply an algorithm that computes a bounding box that contains a series of subsequent points in the FIFO where the distance between the adjacent points in the sorted sequence is less than a predefined length. Then, if the volume of the bounding box is larger than a threshold of becoming a threat, the coordinates of the nearest point in the bounding box are found and used for computing (9). The procedure of the local obstacle map building method proposed above is illustrated in Figure 5.

Experiment Results

In this section, we present the simulation and experiment results of autonomous exploration in an urban environment. The experiment design is carefully scrutinized for the safety regulations; it is performed in a field with portable canopies simulating buildings, not with real ones. The canopies, measuring 3 m × 3 m × 3 m each, are arranged as shown in Figure 6. The distance between one side to the next adjacent side of canopies is set to 10 m in the north-south direction and 12 m in the east-west direction so that the UAV with a 3.5-m-long fuselage can pass

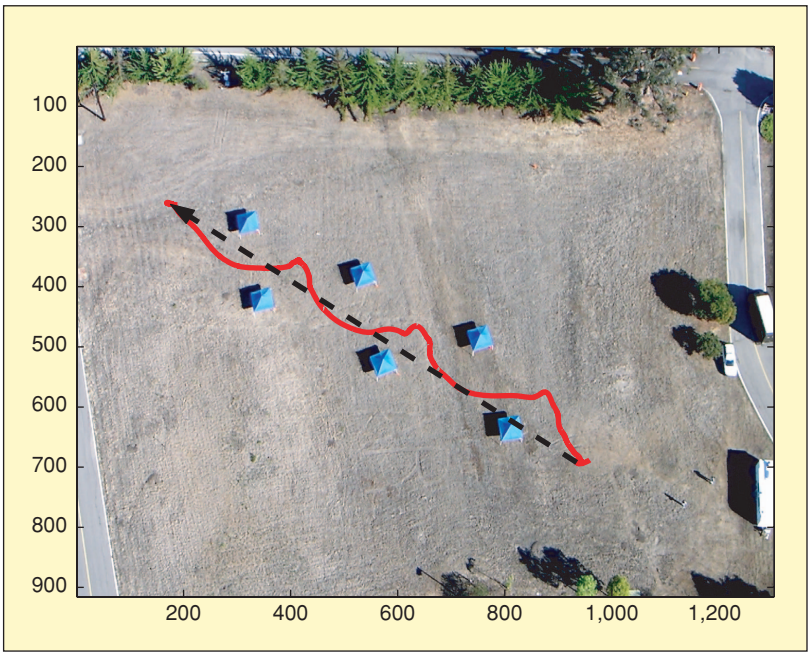

Figure 6. An aerial view of an urban navigation experiment (dashed line: given straight path; solid line: actual flight path of UAV during experiment).

Table 1. Specification of a Berkeley UAV: Ursa Magna 2.

Base platform	Yamaha R-50 Industrial Helicopter
Dimensions (width, length, height)	0.7 m × 3.5 m × 1.08 m
Rotor	Diameter 3.070 m
Weight	44 kg (dry weight) 20 kg (payload including avionics)
Engine	2-cycle gasoline engine 12 hp
Operation time	Fuel: 40 min Avionics: 200 min
Onboard system	CPU: AMD K6 400 MHz PC104 Boeing DQI-NP INS NovAtel GPS MillenRT-2 IEEE 802.11b Wireless Ethernet Ultrasonic altimeters SICK laser range finder (LMS-200)
Capabilities	Preloaded waypoint navigation Interactive waypoint navigation Position tracking mode

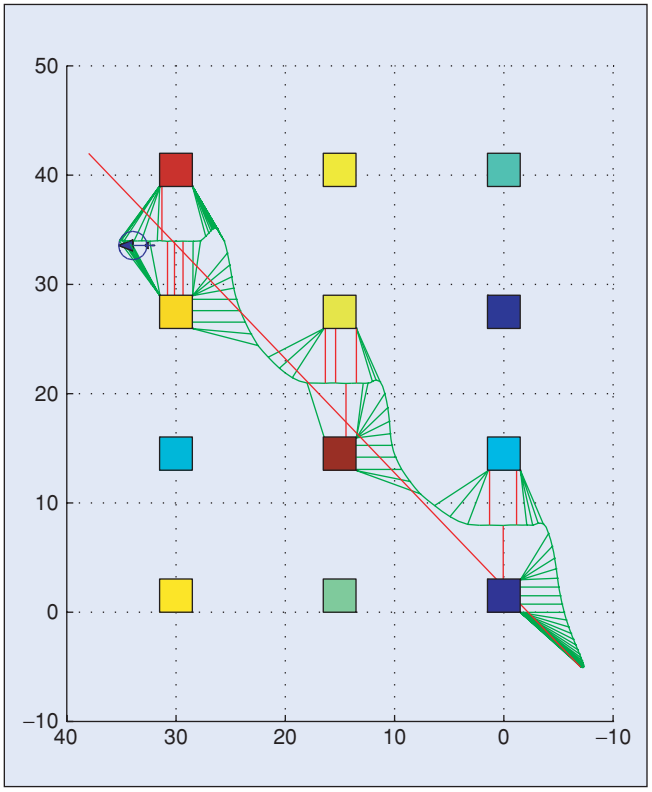

Figure 7. A simulation of MPC-based path planning in the proposed urban experiment setup.

between the canopies with minimal safe clearance, about 3 m from the rotor tip to the nearby canopy when staying on course.

For validation, the MPC engine developed in [4] is applied to the proposed urban experiment setup. A simulation model is constructed in MATLAB/Simulink, where the local map built using a laser scanner is substituted with a prebuilt map. The MPC with the local map building algorithm is implemented in C language for speed and portability. As shown in Figure 7, the MPC path planner demonstrated its capability to generate a collision-free trajectory based on the original trajectory with intentional overlapping with buildings. The lines pointing to the buildings represent $\mathbf{X}_O^{\min}$ computed at each sampling time. For experiments, the Simulink model is modified to function as the online trajectory generator; although Simulink was not designed for a real-time controller in the loop, it can be tweaked to run for soft real-time control by adding a real-time enforcing block. Using the blocking behavior of TCP/IP communication, a custom TCP/IP transport block is configured to enforce soft real-time operation of the Simulink model at 10 Hz in this case.

A series of experiments for urban exploration was performed using a Berkeley UAV (Figure 1), whose detailed specification is given in Table 1. For obstacle detection, the vehicle is equipped with an LMS-200 from Sick AG (Figure 3), a 2-D laser range finder. It has 80 m of detection range with 10-mm resolution. The scanner's measurement is sent to the flight computer via RS-232 and then relayed to the ground station running the MPC-based trajectory generator in Simulink. The trajectory generation module on MATLAB/Simulink and the ground monitoring/commanding software were executed simultaneously on a computer with Pentium 4, 2.4 GHz with 512 MB RAM running Microsoft Windows XP. The laser scanner data is processed following the procedure described above. In Figure 8, a 3-D rendering from the ground station software is presented. The display shows the location of the UAV, the reference point marker, $\mathbf{X}_{O/B}^{\min}$ to a point in the local obstacle map at that moment, and laser-scanned points as dots. During the experiments, the laser scanner was able to detect the canopies in the line of sight with outstanding accuracy as well as other natural and artificial objects, including buildings, trees, and power lines.

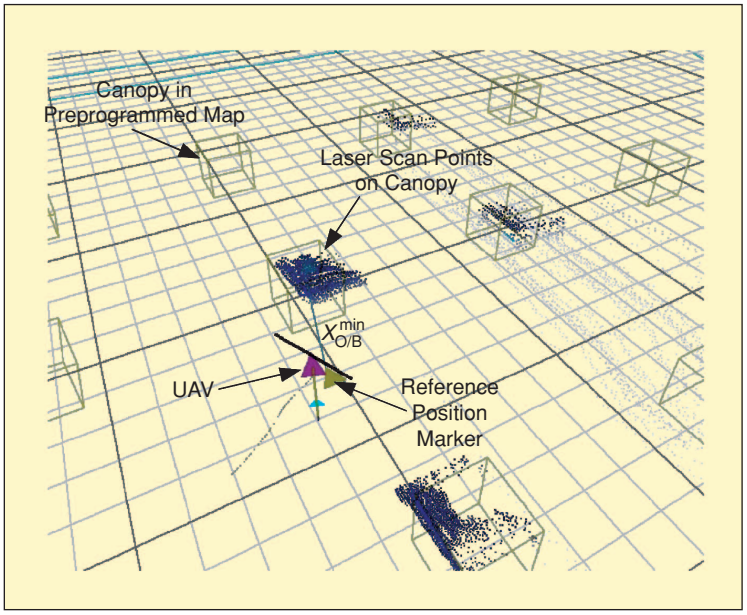

Figure 8. A snapshot of the 3-D rendering during an urban exploration experiment.

The processed laser scanned data in the form of a local obstacle map is used in the optimization (6). The trajectory is then sent via IEEE 802.11b to the onboard flight management system at 10 Hz. The overall system structure used in the experiments is shown in Figure 9. The tracking layer controls the host vehicle to follow the revised trajectory. In the repeated experiments, the vehicle was able to fly around the obstacles with sufficient accuracy for tracking the obstacle-free trajectory, as shown in Figure 6 (solid line).

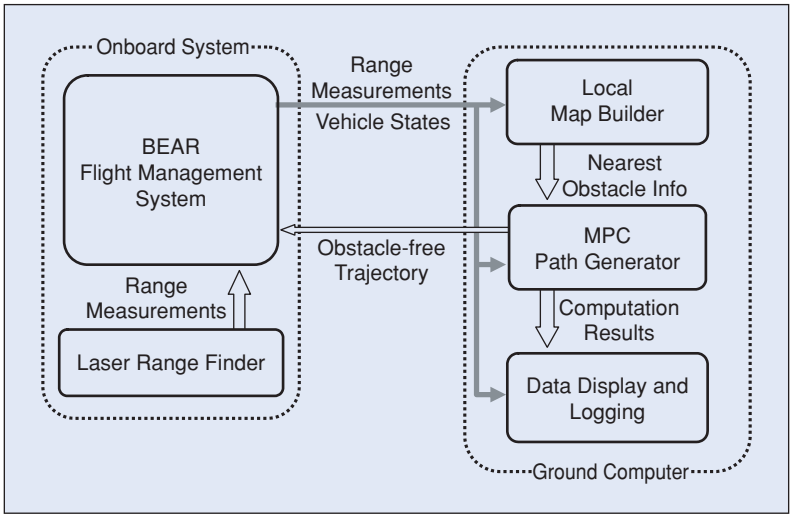

Figure 9. The overall system structure used in the experiments.

Conclusions

This article presented an autonomous exploration method in an unknown environment using MPC-based obstacle avoidance with local map building by onboard sensing. An onboard laser scanner is used to build an online map of obstacles around the vehicle with outstanding accuracy. This local map is combined with a real-time MPC algorithm that generates a safe vehicle path, using a cost function that penalizes the proximity to the nearest obstacle. The adjusted trajectory is then sent to a position tracking layer in the hierarchical UAV avionics architecture. In a series of experiments using a Berkeley UAV, the proposed approach successfully guided the vehicle safely through the urban canyon.

Acknowledgments

This research was supported by the Defense Advanced Research Projects Agency (DARPA) F33615-98-C-3614 and ARO MURI (Army Research Office Multidisciplinary University Research Initiative) DAAD190210383. The authors also wish to thank Travis Pynn and Perry Kavros for their support on the experiments.

Keywords

Unmanned aerial vehicle, obstacle avoidance, urban navigation, laser scanning, model predictive control.

References

[1] S. Thrun, "Robotic mapping: A survey," in *Exploring Artificial Intelligence in the New Millenium*, G. Lakemeyer and B. Nebel, Eds. San Mateo, CA: Morgan Kaufmann, 2002.

[2] M.C. Martin and H.P. Moravec, "Robot evidence grids," CMU Robot. Inst. Tech. Rep. CMU-RI-TR-96-06, Mar. 1996.

[3] G.J. Sutton and R.R. Bitmead, "Computational implementation of NMPC to nonlinear submarine," in *Nonlinear Model Predictive Control, Vol. 26*, F. Allgöwer and A. Zheng, Eds. Cambridge, MA: Birkhäuser, 2000, pp. 461–471.

[4] D.H. Shim, H.J. Kim, and S. Sastry, "Decentralized nonlinear model predictive control of multiple flying robots," in *Proc. IEEE Conf. Decision Control,* Maui, HI, Dec. 2003, pp. 3621–3626.

[5] J. Sprinkle, J.M. Eklund, H.J. Kim, and S. Sastry, "Encoding aerial pursuit/evasion games with fixed wing aircraft into a nonlinear model predictive tracking controller," in *Proc. 43rd IEEE Conf. Decision Control*, Paradise Island, Bahamas, Dec. 2004, pp. 2609–2614.

[6] D.H. Shim, "Hierarchical control system synthesis for rotorcraft-based unmanned aerial vehicles," Ph.D. dissertation, Univ. California, Berkeley, 2000.

[7] R. Moitie and N. Seube, "Guidance algorithms for UUVs obstacle avoidance systems," in *Proc. OCEANS MTS/IEEE Conference and Exhibition*, 2000, vol. 3, pp. 1853–1860.

[8] L. Singh and J. Fuller, "Trajectory generation for a UAV in urban terrain, using nonlinear MPC," in *Proc. American Control Conf.*, 2001, pp. 2301–2308.

[9] W.B. Dunbar, M.B. Milam, R. Franz, and R.M. Murray, "Model predictive control of a thrust-vectored flight control experiment," in *Proc. 15th IFAC World Congress on Automatic Control*, Barcelona, Spain, July 200, pp. 355–360.

[10] J. Bellingham, Y. Kuwata, and J. How, "Stable receding horizon trajectory control for complex environments," in *Proc. AIAA Conf. Guidance, Navigation, Control*, Austin, Texas, Aug. 2003.

David H. Shim received the B.S. and M.S. degrees in mechanical design and production engineering from Seoul National University, Seoul, Korea, and the Ph.D. degree in mechanical engineering from the University of California at Berkeley, in 1991, 1993, and 2000, respectively.

He was with Hyundai Motor Company, Korea, as a transmission design engineer from 1993–1994. In 2001, he joined Maxtor Corporation, California, as a staff servo engineer in charge of advanced servo control system optimization, development, and implementation. He is a principal development engineer at the University of California Berkeley. He is the manager of the Berkeley UAV Research Group, Berkeley Aerobot Team (BEAR). His research interests include advanced flight control, path planning, and multiagent integration for unmanned aerial vehicles.

Hoam Chung received the B.A. and M.S. degree in precision mechanical engineering from Hanyang University, Korea, in 1997 and 1999, respectively, and the Ph.D. degree in mechanical engineering from the University of California, Berkeley, in spring 2006. He is now a specialist at the Electronics Research Laboratory at the University of California, Berkeley.

He has implemented a series of Berkeley Aerobot Team (BEAR) unmanned aerial vehicle (UAV) systems and participated in various UAV projects, including vision-based landing, formation flight, urban exploration, and model predictive control (MPC)-based dynamic path generation. His research interests include decentralized MPC applications in UAV research, rotorcraft dynamics, real-time system design, and robotics.

S. Shankar Sastry was the chair, Department of Electrical Engineering and Computer Sciences, University of California, Berkeley from 2001 to July 2004. He is currently the NEC Distinguished Professor of EECS and Bioengineering. He served as director of the Information Technology Office at Defense Advanced Research Projects Agency (DARPA) from 1999–2001. From 1996–1999, he was the director of the Electronics Research Laboratory at Berkeley, an organized research unit on the Berkeley campus conducting research in computer science and all aspects of electrical engineering. He received his Ph.D. degree in 1981 from the University of California, Berkeley. He was on the faculty of the Massachusetts Institute of Technology (MIT) as an assistant professor from 1980–1982 and Harvard University as a chaired Gordon Mc Kay professor in 1994. He has held visiting appointments at the Australian National University, Canberra the University of Rome, Scuola Normale and University of Pisa, the CNRS laboratory LAAS in Toulouse (poste rouge), Professor Invite at Institut National Polytechnique de Grenoble (CNRS laboratory VERIMAG), and as a Vinton Hayes Visiting fellow at the Center for Intelligent Control Systems at MIT.

Sastry was elected into the National Academy of Engineering in 2001. He also received the President of India Gold Medal in 1977, the IBM Faculty Development award for 1983–1985, the National Science Foundation Presidential Young Investigator Award in 1985, the Eckman Award of the of the American Automatic Control Council in 1990, an M.A. (honoris causa) from Harvard in 1994, Fellow of the IEEE in 1994, the distinguished Alumnus Award of the Indian Institute of Technology in 1999, and the David Marr prize for the best paper at the International Conference in Computer Vision in 1999.

Address for Correspondence: David H. Shim, Department of Electrical Engineering and Computer Science, University of California, Berkeley, California 94720 USA. Phone: +1 510 334 5661. E-mail: hcshim@eecs.berkeley.edu.