2011 50th IEEE Conference on Decision and Control and
European Control Conference (CDC-ECC)
Orlando, FL, USA, December 12-15, 2011

Chain-Based Path Planning for Multiple UAVs

Matthew Argyle, Caleb Chamberlain, and Randy Beard

Abstract— **This paper presents a UAV path planning strategy for optimizing the return over a finite-time horizon, where the return is specified by a bounded differentiable reward function. We represent paths using a simulated chain in a force field, where the forces are influenced by the reward function. The chain adapts continuously to changes in the return function, and produces good paths with minimal computational overhead. We compare the chain-based path planner to a look-ahead planner and extend this approach to multiple UAVs in a centralized manner.**

I. Introduction

Over the years, many applications have been found for Unmanned Aerial Vehicles (UAVs). Two applications that have been the subject of recent focus are target tracking and surveillance. While there have been many approaches presented (See [1], [2], [3], [4], [5]), one issue that always arises, no matter the application, is path planning.

As with many path planning problems, one of the largest challenges inherent in UAV path planning is the size of the configuration space. The number of possible paths is so large that, even with modern computers, we could not hope to search the entire space to find the best path in a reasonable amount of time. For non-holonomic vehicles like a UAV, the problem is made even more interesting because motion constraints (such as minimum turn radius) must not be violated. The result is that, even if we can easily identify a set of positions where we would like the UAV to be, planning the best path to visit them, and in what order, is a non-trivial problem.

One obvious simplification is to restrict the configuration space to a small subset of flyable paths. For example, in [6], Kim restricts UAV paths to orbits with the aircraft's minimum turn radius, and attempts to optimize the center of the orbit so that a ground target is visible as often as possible. Kim discretizes each orbit into a set of positions and tests viewability of the target at each position; the orbit center is adjusted using a simplified hill-climbing algorithm. While Kim's algorithm is reasonably fast, it is restrictive in that it can only plan orbits.

This paper takes a different approach. Instead of optimizing one type of path, this paper introduces a planner that models the UAV path using a series of connected waypoints that serve as links in a simulated chain. The chain is placed inside a force-field that is generated using the gradient of a bounded differentiable reward function, so that each link in the chain tends to move toward local maxima of the reward function. Other forces are applied to prevent UAV flight constraints from being violated. Since the path is represented using waypoints that are a fixed distance apart, it is easy to determine roughly where the UAV will be at any given time. This timing information can be used to prevent collisions and spread out paths when creating plans for multiple UAVs.

The structure of this paper is as follows. Section II develops the dynamics of the chain-based path planner. In Section III we present plans generated by the planner and simulation results comparing the chain-based path planner to an n-step look-ahead planner. In Section IV we extend the chain-based path planner to handle multiple UAVs and explore how it scales to large numbers of UAVs. Section V contains plans generated by the planner for multiple UAVs and a variety of reward functions.

II. Single UAV Chain-Based Path Planner

To generate more detailed, informative UAV paths, we loosely simulate a chain placed in a force-field, where angles between links are constrained to enforce the minimum turn radius of the UAV. The chain model is specifically designed so that good paths can be discovered quickly, and existing paths can be modified in real-time as the bounded differentiable reward function changes.

Similar to [7], we model the chain as a collection of unit-mass points constrained to the 2-D plane. Letting $\mathbf{z}_i = (x_i, y_i)$ be the position of the ith element in the chain, then a N-link chain is represented by

$$\mathbf{c} = [\, \mathbf{z}_1\, \mathbf{z}_2\, \ldots\, \mathbf{z}_N \,]^T .$$

Let $\mathcal{F}(\mathbf{z})$ be a bounded differentiable function. Without loss of generality, assume that $\mathcal{F}$ is normalized so that the minimum is zero and the maximum is one. The magnitude of the force applied to link i is proportional to $(1 - \mathcal{F}(\mathbf{z}_i))$, while the direction of the force is given by the gradient of $\mathcal{F}(\mathbf{z}_i)$. Let $\mathbf{g}_i$ be the unit vector pointing in the direction of the gradient of $\mathcal{F}(\mathbf{z}_i)$, i.e.

$$\mathbf{g}_i = \frac{\partial \mathcal{F}(\mathbf{z}_i)}{\partial \mathbf{z}_i} \Big/ \left\| \frac{\partial \mathcal{F}(\mathbf{z}_i)}{\partial \mathbf{z}_i} \right\| .$$

The unconstrained dynamics for link i are then

$$\ddot{\mathbf{z}}_i = \gamma_1 (1 - \mathcal{F}(\mathbf{z}_i)) \mathbf{g}_i ,$$

where γ_1 is a positive constant. The unconstrained dynamics of the entire chain are given by

$$\ddot{\mathbf{c}} = \gamma_1 \mathbf{u} \qquad (1)$$

M. Argyle is a PhD student with the Department of Electrical Eng., Brigham Young University, Provo, Utah, 84602 matt.argyle@gmail.com
C. Chamberlain is with the Ch Robotics L.L.C. Provo, Utah 84606 voiceafx@gmail.com
R. Beard is with the Department of Electrical Eng., Brigham Young University, Provo, Utah, 84601 beard@byu.edu

where

$$\mathbf{u} = \begin{bmatrix} (1-\mathcal{F}(\mathbf{z}_1))\mathbf{g}_1 \\ (1-\mathcal{F}(\mathbf{z}_2))\mathbf{g}_2 \\ \vdots \\ (1-\mathcal{F}(\mathbf{z}_N))\mathbf{g}_N \end{bmatrix} \tag{2}$$

We constrain link motion so that the distance between adjacent links is kept constant. Let L be the desired distance between each adjacent chain link, and define the constraint vector $\phi(\mathbf{c})$ as

$$\phi(\mathbf{c}) \triangleq \begin{bmatrix} \|\mathbf{z}_2-\mathbf{z}_1\|^2 - L^2 \\ \|\mathbf{z}_3-\mathbf{z}_2\|^2 - L^2 \\ \vdots \\ \|\mathbf{z}_N-\mathbf{z}_{N-1}\|^2 - L^2 \end{bmatrix}. \tag{3}$$

As long as each element of $\phi(\mathbf{c})$ is zero, the distances between each link in the chain will be correct. To push each element of $\phi(\mathbf{c})$ toward zero, we add a restoring term to Equation (1) to obtain

$$\ddot{\mathbf{c}} = \gamma_1\mathbf{u} - \gamma_2\frac{\partial\phi}{\partial\mathbf{c}}\phi,$$

where γ_2 is a positive constant. Large values of γ_2 force adjacent links to remain a distance L apart. On the other hand, as γ_2 gets large, the differential equation stiffens, requiring more computational resources to simulate the model accurately. A tradeoff must be made between maintaining exact distances, and modeling the chain movement in a computationally efficient manner.

To ensure that the minimum turn radius of the UAV is not violated, straightening forces are applied to each link in the chain. Let r_{min} be the minimum turn radius of the UAV and let θ_{max} be the maximum allowable angle formed by the vectors between three adjacent links. Then the minimum number of links required to complete a full circle is given by

$$n = \frac{2\pi}{\theta_{max}}. \tag{4}$$

As shown in Figure 1, the length of the approximate circular path is at least as long as a circle with radius r_{min}, or

$$nL \geq 2\pi r_{min}. \tag{5}$$

Combining Equations (4) and (5), we get

$$\theta_{max} \leq \frac{L}{r_{min}}. \tag{6}$$

To guarantee that turn radius constraints are not violated, Equation (6) is treated as a strict equality.

As shown in Figure 2, the straightening force applied to link i is designed to ensure that $|\theta_i| < \theta_{max}$, where θ_i is the angle between $\mathbf{v}_i^1$ and $\mathbf{v}_i^2$, defined as

$$\mathbf{v}_i^1 = (\mathbf{z}_i - \mathbf{z}_{i-1})/\|\mathbf{z}_i - \mathbf{z}_{i-1}\|,$$
$$\mathbf{v}_i^2 = (\mathbf{z}_{i-1} - \mathbf{z}_{i-2})/\|\mathbf{z}_{i-1} - \mathbf{z}_{i-2}\|$$

It follows that θ_i is given by

$$\theta_i = \arccos(\mathbf{v}_i^1 \cdot \mathbf{v}_i^2).$$

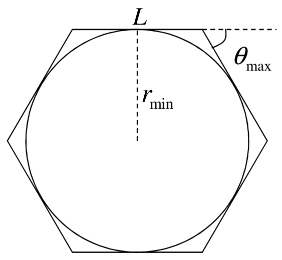

Fig. 1. Definition of θ_{max}, the maximum allowable turn angle to approximate a circle using a discrete chain.

The straightening force for link i is given by

$$\mathbf{f}_i = \frac{\lambda_i}{1+\exp(k(\theta_{max}-\theta_i))}(\mathbf{v}_i^1)^\perp$$

where k is a positive constant that defines how closely the logistic function approximates a step function, λ_i is the upper limit of the straightening force for link i, and

$$(\mathbf{v}_i^1)^\perp = \begin{pmatrix} 0 & -1 \\ 1 & 0 \end{pmatrix}\mathbf{v}_i^1.$$

For each link, λ_i must be large enough to at least match the sum of all possible forces on all subsequent links in the chain; otherwise, there may be cases where the straightening force might not be high enough to prevent minimum turn radius constraints from being violated. If there are N links in the chain, and $i = 1$ corresponds to the first link, then we define λ_i as

$$\lambda_i = \gamma_1(N+1-i).$$

Recall that γ_1 is the largest possible magnitude of the unconstrained force applied by the force-field to each link.

There is a tradeoff that must be taken into account when selecting the constant k. Ideally the straightening force would only be applied when the maximum angle constraint is violated, and then only enough to correct the violation. However, this is computationally infeasible. Large values of k, which would approximate a step function, cause the differential equations to become stiff and require more computing power. On the other hand, the smaller k is, the sooner the straightening force becomes significant which limits the turns the links can make.

Let $\mathbf{f}_s$ be the vector of all straightening forces applied to the chain, defined as

$$\mathbf{f}_s \triangleq [\,\mathbf{f}_1\,\mathbf{f}_2\,\cdots\mathbf{f}_N\,]^T. \tag{7}$$

Then, the chain dynamics with straightening forces are given by

$$\ddot{\mathbf{c}} = \gamma_1\mathbf{u} - \gamma_2\frac{\partial\phi}{\partial\mathbf{c}}\phi + \mathbf{f}_s.$$

A damping term is added to reduce oscillations to obtain

$$\ddot{\mathbf{c}} = \gamma_1\mathbf{u} - \gamma_2\frac{\partial\phi}{\partial\mathbf{c}}\phi + \mathbf{f}_s - \gamma_3\dot{\mathbf{c}} \tag{8}$$

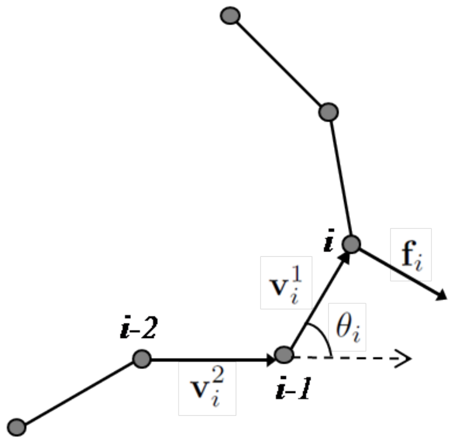

Fig. 2. Definition of terms for chain-based path planner.

where γ_3 is a positive constant.

When using the chain to plan paths in real-time, the first two links serve as waypoints for the UAV to follow, and are not allowed to move. When the UAV nears the end of the first link, the link is removed from the chain and a new link is added to the end, and the two links that then comprise the beginning of the chain are fixed as new waypoints. Therefore the UAV always has two unchanging waypoints to follow while the remainder of the chain continuously adapts to changing target conditions.

One issue that occurs is that the straight line links between the nodes of the chain might not be flyable. The equal length trajectory smoothing technique described in [8] can be used to provide flyable paths while still maintaining the timing of each of the later waypoints.

III. Single UAV Simulation Results

Figure 3 shows a number of paths generated by the chain-based planner in realistic situations. The reward function was generated by estimating, by a mixture of Gaussians, the probability of detecting a target within a city where the buildings partially occlude the target.

In order to test the effectiveness of the chain-based path planner, we created a simulation consisting of a 6-DOF model of a UAV flying within a 300 by 300 meter world. The UAV had a cruising speed of 10 m/s and a minimum turn radius of 20 meters. Three different reward functions were used: a 2D gaussian

$$f_3(x, y) = \alpha e^{-\frac{1}{2\sigma^2}\left(x^2+y^2\right)}, \tag{9}$$

where $\sigma^2 = 50$ is the variance and α was used to normalize the reward function to have a maximum value of one. The second reward function is a 'doughnut'

$$f(x, y) = \alpha e^{-\nu 8\left|(\sqrt{x^2+y^2}-R)\right|} \tag{10}$$

where $\nu = 0.01$ controls the drop off of the curve, $R = 100$ is the radius of the doughnut, and α normalizes the function.

TABLE I

Percent difference in score between planned path from the chain-based path planner and the actual path the UAV flew.

Function	Doughnut	Gaussian	Mixture of Gaussians
Mean (%)	-0.86	-4.2	-9.37
Standard Deviation (%)	0.39	3.91	13.36

The final reward function is a mixture of three Gaussians

$$f(x, y) = \alpha \sum_{i=1}^{3} \beta_i \left(2\pi\right)^{-\frac{3}{2}} \left|\Sigma_i\right|^{-\frac{1}{2}} e^{-\frac{1}{2}(\mathbf{c}-\mu_i)^T \Sigma_i^{-1}(\mathbf{c}-\mu_i)} \tag{11}$$

where α scales the reward function to have a maximum of one, $0 < \beta_i < 1$ is a random positive weight of the ith gaussian, μ_i is a random mean for the ith gaussian, and Σ_i is a random covariance matrix for the ith gaussian.

One hundred Monte Carlo simulations were run for each reward function with random UAV starting position and orientation for the 2D Gaussian and doughnut reward functions and random UAV starting position and orientation as well as random means, covariances, and weights for the mixture of Gaussian reward function. For each simulation a chain-based path planner, with twelve links for the 2D Gaussian and doughnut and eight links for the mixture of Gaussians and a link length of 20 meters, and a 5-step look-ahead planner, see [9] for a good overview, consisting of straight paths and the following possible heading changes $\theta \in \{-20, -10, -5, 0, 5, 10, 20\}$. Each simulation ran for 100 seconds of flight time and the score of the UAVs path, defined as the integral of the reward function over the UAVs path, was recorded for both path planners.

Table I shows the percentage difference in score between the planned path and the path that the UAV actually flew. As can be seen, the UAV was not able to perfectly fly the path generated by the chain-based path planner due to the instantaneous turns required. The error is larger for the Gaussian and mixture of Gaussians functions than the doughnut because of the numerous sharp terms being performed. However, the path that was flown is comparable to the desired path.

Table II shows the percentage difference in score between the planned path from the chain-based planner and the look-ahead planner. Notice that, on average, the chain-based path planner is comparable to the look ahead planner for all three functions. However, data from one of the mixture of Gaussians scenarios was not used when computing the mean and standard deviation because it was a severe outlier. In this particular scenario, the chain-based path planner did 980% better than the look ahead planner. This occurred because the UAV started in a position that the look-ahead planner could not find any of the gaussian distribution maxima. The chain-based path planner had no difficulty finding these maxima due to the much longer chain length than the look-ahead length. If this scenario was included in the data, then the mean would have been 16.54% and the standard deviation would be 100.32%.

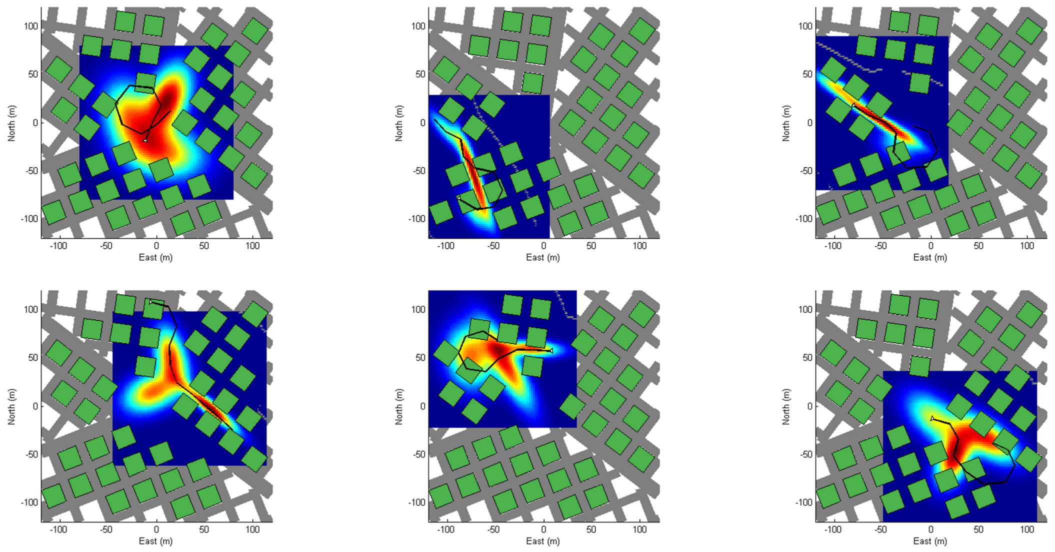

Fig. 3. Examples of paths generated by chain-based path planner for a target surveillance application

TABLE II

PERCENT DIFFERENCE IN SCORE BETWEEN THE PATH FROM THE CHAIN-BASED PATH PLANNER AND 5-STEP LOOK-AHEAD PLANNER. A NEGATIVE VALUE MEANS THE LOOK-AHEAD PLANNER PERFORMED BETTER THAN THE CHAIN-BASED PLANNER.

Function	Doughnut	Gaussian	Mixture of Gaussians
Mean (%)	-0.45	-0.95	6.8096
Standard Deviation (%)	2.65	2.255	17.286

IV. MULTIPLE UAVS CHAIN-BASED PATH PLANNER

There are several possible methods for extending the single UAV chain-based path planner to multiple UAVs. The simplest method is for each UAV to have its own independent chain. While this approach has several benefits, such as the UAVs do not need to communicate, there are several disadvantages. First, if the UAVs are flying at the same altitude, independent chains will do nothing to prevent collisions between the UAVs. Second, if the UAVs start in similar locations they could fly almost identical paths. This would provide little benefit in most surveillance or tracking applications. Instead, we will develop a method in which each UAV has its own chain but the chains interact with each other.

Let the jth UAV have an N-link chain $\mathbf{c}_j = [\mathbf{z}_{j1}, \mathbf{z}_{j2}, \ldots, \mathbf{z}_{jN}]^T$ We will start with the single UAV chain dynamics given in Equation (8). Note that each UAV can have a different minimum turn radius. In addition, the desired distance between links, L_j, does not need to be the same for each UAV. However, L_j should be proportional to the speed of the jth UAV. This restriction implies that each UAV will fly through the ith node in its chain at approximately the same time.

We create another force that causes the chains of different UAVs to repel each other. Let $\mathbf{d}_{ij}(m,n)$ be the vector going from the nth node of the jth UAV to the mth node of the ith UAV, or

$$\mathbf{d}_{ij}(m,n) = \mathbf{z}_{im} - \mathbf{z}_{jn}$$

and

$$\hat{\mathbf{d}}_{ij}(m,n) = \frac{\mathbf{d}_{ij}(m,n)}{\|\mathbf{d}_{ij}(m,n)\|}.$$

Define the repulsive force acting on node m of the ith UAV due to node n of the jth UAV to be

$$r_{ij}(m,n) = \begin{cases} \hat{\mathbf{d}}_{ij}(m,n)\gamma_{ij4}e^{-\gamma_{ij5}\|\mathbf{d}_{ij}(m,n)\|} & \text{if } |m-n| < k \text{ and } \\ & \|\mathbf{d}_{ij}(m,n)\| < d_{max} \\ 0 & \text{otherwise} \end{cases} \quad (12)$$

where γ_{ij4} is a positive constant that scales the repulsive force, γ_{ij5} is a positive constant that controls how quickly the force drops off due to distance, k is a positive integer that determines how time dependent the interactions are, and d_{max} is the furthest distance that nodes can influence each other from. By not requiring that $\gamma_{ij4} = \gamma_{ji4}$ and $\gamma_{ij5} = \gamma_{ji5}$, UAVs can be given precedence over others. For example, if $\gamma_{ij4} > \gamma_{ji4} = 0$ and $\gamma_{ij5} > \gamma_{ji5} = 0$ then the ith UAV's path would be pushed away from the jth UAV's path while the jth UAV's path is not influenced by the ith UAV's path.

Merging all of the repulsive forces acting on the chain of the ith UAV by the chain of the jth UAV, we obtain

$$\mathbf{R}_{ij} = [R_{ij}(1)\; R_{ij}(2)\; \ldots\; R_{ij}(N)]^T$$

where

$$R_{ij}(n) = \sum_{m=1}^{N} r_{ij}(n,m)$$

is the sum of all the forces acting on the nth node of the ith UAV by all the nodes of the jth chain.

Adding the total repulsive forces to Equation (8) we obtain the complete chain dynamics for the jth UAV

$$\ddot{\mathbf{c}}_j = \gamma_{j1}\mathbf{u}_j - \gamma_{j2}\frac{\partial \phi_j}{\partial \mathbf{c}_j}\phi_j + \mathbf{f}_{js} - \gamma_{j3}\dot{\mathbf{c}} + \sum_{i \neq j}\mathbf{R}_{ji}. \quad (13)$$

As it is currently implemented, the chain-based path planning method for multiple UAVs is unsuited for a decentralized implementation. This is because every agent needs the link locations for every other agent in order to generate the repulsive forces. Even if the planner is implemented in a centralized manner, good communication is required from the planner to all of the agents due to waypoints needing to be sent every couple of seconds.

A. Scaling

Over the last few years, there has been a lot of focus on large groups of UAVs. Path planners for these teams need to scale to the large number of agents well. The non-repulsive portion of the multi-UAV chain-based planner obviously scales on the order of N_u where N_u is the number of UAVs. However, as will be shown, the repulsive forces do not scale as well.

In the worst case, $k = \infty$ and $d_{max} = \infty$, where k and d_{max} are as defined in Equation (12). In this case each link in the chain interacts, or is repelled, with $(N_u - 1)N$ other links where N is the number of links in a UAV chain. There are a total of $N_u N$ links which means the total number of repulsive forces is

$$N_{forces} = (N_u - 1)N_u N^2.$$

Therefore in the worst case, the multiple UAV chain-based planner scales on the order of N_u^2.

If $k < \infty$, the total number of repulsive forces can drop drastically. In this case, each link in the chain interacts with at most $(N_u - 1)(2k - 1)$ other links providing a total number of repulsive forces of

$$N_{forces} = (N_u - 1)N_u(2k - 1)N.$$

This number can be lowered much further if $d_{max} < \infty$, however it will still scale on the order of N_u^2.

V. Multiple UAV Simulation Results

Figure 4 shows how the generated paths can vary by modifying γ_{ij4} and γ_{ij5}. In the first two images, only the black path is being repulsed ($\gamma_{124} \neq 0$ and $\gamma_{215} = 0$). In the last image, both chains are repelling each other. Figure 5 shows several paths generated with different UAV and target starting locations. In these images, the repulsive parameters do not change.

VI. Conclusions and Future Work

In this paper, we presented a UAV path planner based on a chain in a force field. This path planner can operate for a single UAV or, with the addition of repulsive forces, multiple UAVs. This planner tries to find paths that go through maxima of an underlying bounded differentiable reward function.

The chain-based planner is advantageous because it runs quickly, paths can be modified continuously as the underlying differentiable function changes, and there are few restrictions on the shape of the generated path. In addition, because the paths are continuously updated, there is no required amount of precessing time to compute a flyable path. At any given moment, the path is valid. Giving the chain-based path planner more processing time allows a larger chain to be computed or more time spent simulating the chain.

We currently have several areas to explore in the future. First, we want to extend this to three dimensions. While it appears that the extension will be very straight forward, we will need to see if modern processors will be able to handle the additional computational requirements. Second, we would like to apply this planner to a target tracking problem. Because the chain-based planner encodes the expected UAV position with respect to time, allowing each UAV to determine when it expects its line of sight to be occluded, it might be possible to weight the forces on each chain link so that multiple UAVs can ”hand-off” target tracking responsibility so that at least one sensor is positioned to view the ground target at all times. Third, we would like to modify the planner so that each UAV has multiple chains initialized in different directions. The UAV would then fly to the next waypoint along the path that has the best fitness. As it flies to the next waypoint, it would remove all other chains and initialize new ones. This should increase performance by allowing the planner to explore more of the space while not requiring much more computational power. Finally, we want to develop a decentralized version of the planner.

VII. Acknowledgments

This research was supported by OSD and AFRL under contract FA8650-08-C-1411 as a subcontract SET-2008-0026 from SET to BYU.

References

[1] Z. Tang and U. Ozguner, “Sensor fusion for target track maintenance with multiple UAVs based on Bayesian filtering method and hospitability map,” in *Proc. 42nd IEEE Conference on Decision and Control*, vol. 1, 9–12 Dec. 2003, pp. 19–24.

[2] U. Zengin and A. Dogan, “Real-time target tracking for autonomous UAVs in adversarial environments: A gradient search algorithm,” vol. 23, no. 2, pp. 294–307, April 2007.

[3] “On optimal sensor placement and motion coordination for target tracking,” in *Proc. IEEE International Conference on Robotics and Automation ICRA 2005*, 18–22 April 2005, pp. 4544–4549.

[4] S. Kanchanavally, R. Ordonez, and J. Layne, “Mobile target tracking by networked uninhabited autonomous vehicles via hospitability maps,” in *Proc. American Control Conference the 2004*, vol. 6, 30 June–2 July 2004, pp. 5570–5575.

[5] B. Grocholsky, J. Keller, V. Kumar, and G. Pappas, “Cooperative air and ground surveillance,” *IEEE Robotics & Automation Magazine*, vol. 13, no. 3, pp. 16–25, Sept. 2006.

[6] J. K. Y. Kim, “Moving target tracking in dense obstacle areas using UAVs,” in *Proc. 17th IFAC World Congress, Seoul, Korea*, 2008.

[7] T. W. McLain and R. W. Beard, “Trajectory planning for coordinated rendezvous of unmanned air vehicles,” in *Proc. GNC 2000*, 2000, pp. 1247–1254.

[8] E. P. Anderson, R. W. Beard, and T. W. Mclain, “Real Time Dynamic Trajectory Smoothing for Uninhabited Aerial Vehicles,” *IEEE Transaction on Control Systems Technology*, vol. 13, pp. 471–477, 2005.

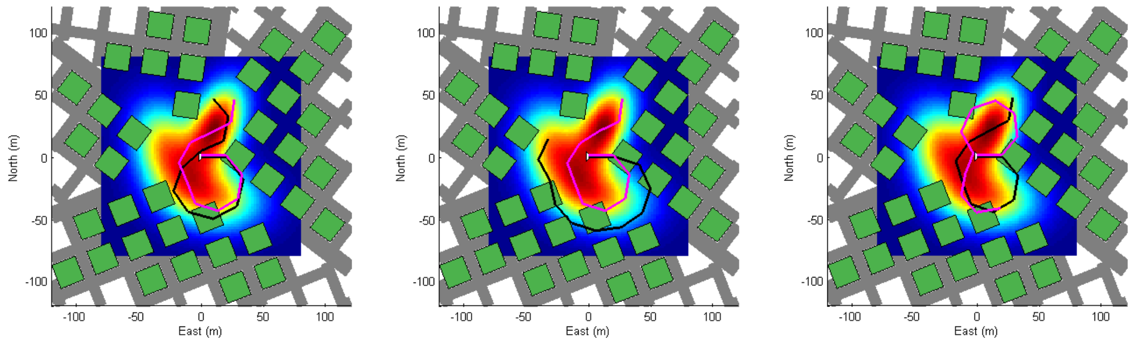

Fig. 4. Examples of paths generated by the chain-based path planner as the repulsive parameters change

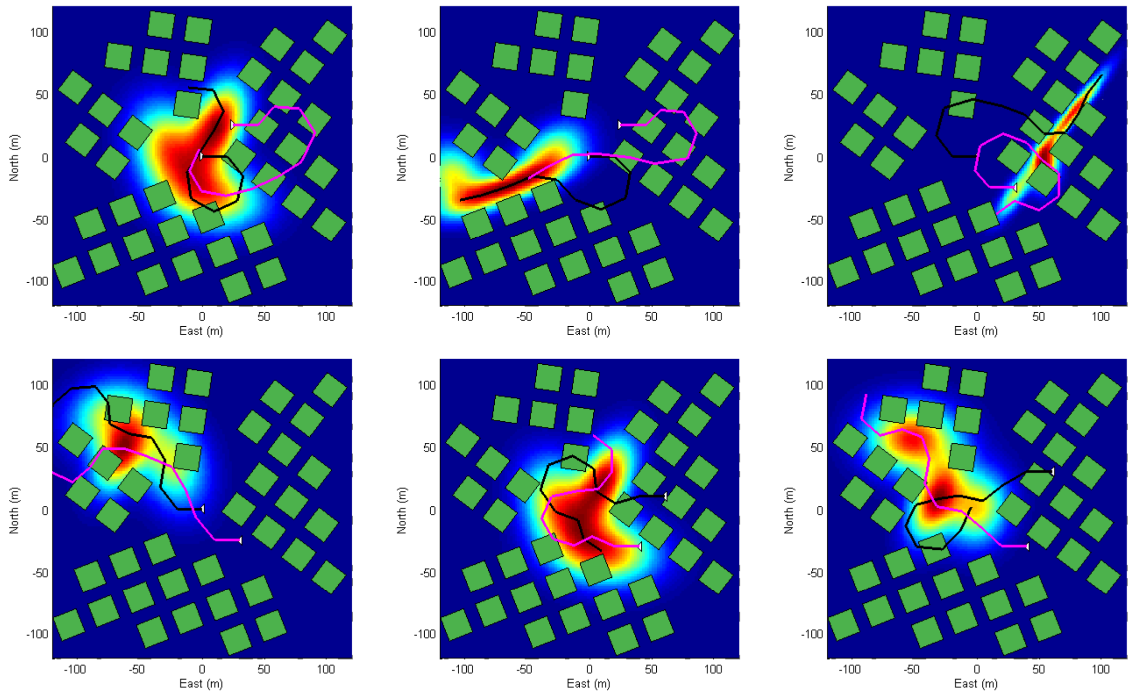

Fig. 5. Examples of paths generated by the chain-based path planner as the repulsive parameters stay constant

[9] P. Niedfeldt, R. Beard, B. Morse, and S. Pledgie, "Integrated Sensor Guidance using Probability of Object Identification," in *Proceedings of the American Control Conference*, 2010, pp. 788–793.