Foresight: Remote Sensing For Autonomous Vehicles Using a Small Unmanned Aerial Vehicle

Alex Wallar, Brandon Araki, Raphael Chang, Javier Alonso-Mora, and Daniela Rus

Abstract A large number of traffic accidents, especially those involving vulnerable road users such as pedestrians and cyclists, are due to blind spots for the driver, for example when a vehicle takes a turn with poor visibility or when a pedestrian crosses from behind a parked vehicle. In these accidents, the consequences for the vulnerable road users are dramatic. Autonomous cars have the potential to drastically reduce traffic accidents thanks to high-performance sensing and reasoning. However, their perception capabilities are still limited to the field of view of their sensors. We propose to extend the perception capabilities of a vehicle, autonomous or human-driven, with a small Unmanned Aerial Vehicle (UAV) capable of taking off from the car, flying around corners to gather additional data from blind spots and landing back on the car after a mission. We present a holistic framework to detect blind spots in the map that is built by the car, plan an informative path for the drone, and detect potential threats occluded to the car. We have tested our approach with an autonomous car equipped with a drone.

1 Introduction

Globally, over 3000 people lose their lives in vehicle-related accidents and over one hundred thousand are injured or disabled on average *every day* [4]. In the United

Alex Wallar, Brandon Araki, Raphael Chang, Daniela Rus
Distributed Robotics Lab, CSAIL, Massachusetts Institute of Technology
32-376 Vassar St, Cambridge, MA, 02139, USA
e-mail: {wallar,araki,rus}@csail.mit.edu
e-mail: raphaelchang@mit.edu

Javier Alonso-Mora
Delft Center for Systems and Control, Delft University of Technology
Mekelweg 2, Room C-2-310, 2628 CD Delft, Netherlands
e-mail: j.alonsomora@tudelft.nl

States, over 90% of these accidents are due to human error [1]. This has resulted in the continued development of advanced safety systems by commercial car manufacturers. For example, systems exist to automatically brake in the case of unexpected obstacles [18], maintain a car in a lane at a given speed [6], and alert users of pedestrians, signage, and other vehicles on the roadway [7]. These systems will make our cars safer and eventually autonomous. However, many accidents are due to blind spots, for example when a vehicle takes a turn with poor visibility or when a pedestrian crosses from behind a parked vehicle. In these accidents, vulnerable road users, i.e. pedestrians and bikers, are typically involved and the consequences are dramatic.

We propose to extend the perception capabilities of a vehicle, autonomous or human-driven, with a small Unmanned Aerial Vehicle (UAV) capable of taking off from the car, flying around corners to gather additional data from blind spots and landing back on the car after a mission. Small UAVs are highly mobile and agile, and they are capable of capturing aerial footage autonomously[17]. The quadcopter could use the car as a charging base, while the car could send the quadcopter out on missions to scout ahead and fill in the blind spots in its vision.

A crucial step in enabling a small UAV and an autonomous car to work together is to ensure that there exists an accurate pose transform between the car and the quadcopter. In this paper, we present a method for relative localization using ultra-wideband radios (UWBs) to measure the relative position of the quadcopter relative to the car with an accuracy of less than 14cm. This information is fused with the internal state estimation to enable the UAV to safely navigate to blind spots and then land back on the car. Furthermore, we have developed a path planning algorithm for remote sensing and have carried out experiments using a quadrotor and a car.

Our method has similarities with multi-robot mapping. For instance [15] and [9] combined the maps from a ground and an aerial robot for enhanced exploration and [8] employed a map created by an aerial robot for planning the motion of a ground robot.

Quadrotors have been able to track a moving vehicle using visual techniques such as AprilTag localization [5], optical flow [11], and infrared markers [22]. However, a weakness of visual systems is that visual cues must be in the line of sight of the quadrotor-mounted camera. Moreover, the lighting conditions must be suitable for the cameras. These restrictions limit the usefulness of visual tracking. Meanwhile, GPS tracking, while useful in long-range outdoors scenarios, is not accurate enough for maneuvers such as landing and obstacle avoidance. For example, average accuracy in smartphone GPS receivers is 4.9 m [2].

UWB sensors have in recent years become a popular tool for localization of quadrotors, particularly in indoor GPS-denied environments [14, 19, 12], since they can provide distance measurements with an accuracy of 10 cm and a range of up to 300 m[3]. Indoor localization of quadrotors has been achieved by placing UWB "anchors" around the perimeter of a room and attaching a UWB "tag" on the quadrotor. Using this technique, [16] and [13] have achieved localization with accuracy on the order of 30cm.

The problem of relative localization is more challenging because the UWB tag is outside of the perimeter of the anchors. However, in [10], a quadrotor outfitted with four UWBs was able to follow a person carrying a UWB tag in the plane of the quadrotor with less than 10 cm mean error. The authors achieved this accuracy using an iterated Extended Kalman Filter. Building off of this work, we used 6 UWB sensors on a car to estimate the 3D position of a UWB on a quadrotor with an average mean error of 13.7 cm. Moreover, unlike in previous systems, we take advantage of the accurate relative transform to use the sensors on the car to plan safe paths for the quadrotor.

1.1 Contribution

This paper presents a method for extending the sensing capabilities of self-driving vehicles by using a small quadrotor to autonomously locate and observe regions occluded to the vehicle and detect potentially unsafe obstacles such as pedestrians or other cars. Our contributions include

- A method for determining the relative transformation between a ground vehicle and a quadrotor using an array of ultra-wideband radios
- An informative planning algorithm that computes collision free paths for the quadrotor relative to the ground vehicle that view occluded regions
- A system that uses the localization and planning algorithms and enables a UAV to position itself and transmit images outside the field of view of the sensors on the car
- Experimental validation using a sensor-equipped Toyota Prius and a Parrot Bebop 2 quadrotor

1.2 Method overview

We consider two vehicles.

- A ground vehicle, i.e. the car, which can create a local map of the environment, localize with respect to it and autonomously navigate. We utilize a 2D grid map to represent the free space and obstacles seen by the car. In particular, our vehicle is equipped with a 2D LIDAR.
- A lightweight companion quadrotor equipped with a front facing camera. The drone is able to fly autonomously to/from the ground vehicle and detect obstacles that were originally occluded for the ground vehicle.

Given a laser scan from the ground vehicle, our objective is to: a) determine which areas of the environment are occluded to ground vehicle, b) compute a safe path for the aerial vehicle to observe the occluded areas, and c) detect unseen obstacles, such as pedestrians, and report them back to the ground vehicle. The quadrotor will take off from the vehicle when it is ready to begin driving and land back on the vehicle when it has parked.

To accurately localize the quadrotor relative to the ground vehicle, we equip the ground vehicle with several UWBs. In the quadrotor, we fuse, via an Unscented Kalman filter (UKF) relative information from a UWB radio with odometry estimates from a down-facing optical flow sensor and an onboard IMU. Our algorithm operates directly on the laser scan from the ground vehicle to find occluded regions. We then employ an anytime sampling-based algorithm to compute a collision free path for the drone that maximizes the occluded area viewed by the quadrotor. To detect obstacles within the occluded areas, we employ a real-time object detecting convolutional neural network [20], which is able to classify and locate objects, such as pedestrians, cars, bicycles, in monocular images. These obstacles are then reported back to the ground vehicle.

Using the ground vehicle's 2D laser scan, we compute the areas it is unable to sense. We employ an anytime sampling-based algorithm to construct a collision free path for the quadrotor that maximizes the total area of the occluded regions it is able to observe. While the quadrotor is executing the planned path, we use a convolutional neural network to classify and detect objects in the quadrotor's field of view, such as pedestrians and cars, and relay this information back to the ground vehicle. The driver of the vehicle is then able to view the quadrotor's camera feed along with the annotated objects. The path is updated if it is no longer collision free due to changes in the laser scan or if a new path is computed that can observe a larger occluded area. A high level overview of the entire process is shown in Algo. 1.

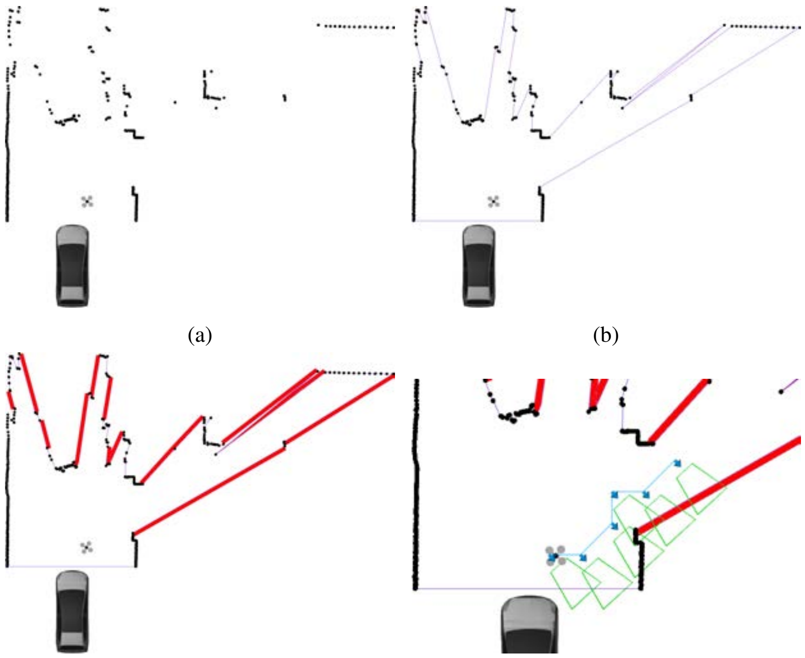
Algorithm 1 Overview of the Foresight algorithm

```
$\Pi \leftarrow \emptyset$
while IsRunning() do
    $x \leftarrow$ GetQuadrotorConfiguration()
    $L \leftarrow$ GetLaserScan()
    $\mathcal{P} \leftarrow$ ConstructBoundingPolygon($L$)
    $\mathcal{B} \leftarrow$ ComputeBlindRegions($L$)
    $\widetilde{\Pi} \leftarrow$ ComputeCoveragePath($x, \mathcal{B}, \mathcal{P}$)
    if $|\Pi| = 0 \lor \neg$PathCollisionFree($\Pi, \mathcal{P}$)$\lor$
        ObservingArea($\widetilde{\Pi}, \mathcal{B}$) $>$ ObservingArea($\Pi, \mathcal{B}$) then
        $\Pi \leftarrow \widetilde{\Pi}$
    SendPathToQuadrotor($\Pi$)
```

2 Planning For Exploration

Planning a path to observe the blind spots of an autonomous car is broken into following steps. First, using the 2D laser scan from the car, we compute a bounding polygon. This polygon represents the known free space where the quadrotor can travel. We use the laser scan to determine regions in space where that the car is not able to sense. These regions are called blind regions. A path is then computed for the quadrotor that maximizes the observed area of the blind regions while staying within the bounding polygon for a given time horizon.

The remainder of this section is structured as follows; Sec. 2.1 introduces the our formal definition of a laser scan and describes how the bounding polygon is found, Sec. 2.2 describes how the blind regions are computed from the laser scan, and Sec. 2.3 describes the algorithm we developed for computing the exploratory path.



(a)

(b)

(c)

(d)

Fig. 1: Plots showing the four stages of the planner. Fig. (a) shows the points from the laser scan. Fig. (b) shows bounding polygon created from the laser scan. Fig. (c) shows the regions occluded to the vehicle in red and Fig. (d) shows the initial plan for the quadrotor to view some of these blind regions.

2.1 Finding the Bounding Polygon

The bounding polygon computed using a scan from the 2D LiDAR sensor on the car is used as a conservative representation of the free space in which the quadrotor can travel. Below we provide a formal definition of a laser scan that is used in the rest of the paper.

Definition 1. A laser scan is a sequence of points, $L = \{\mathbf{c} + r_i \cdot [\cos\theta_i, \sin\theta_i]^T : \theta_{\min} \leq \theta_i \leq \theta_{\max}\} \subset \mathbb{R}^2$, where $\mathbf{c}$ is the 2D position of the LiDAR sensor, r_i is the distance from the sensor to the closest obstruction in the θ_i direction, and $[\theta_{\min}, \theta_{\max}]$ is the angular range of the sensor.

From the laser scan, we compute a bounding polygon. The bounding polygon is defined as the minimum area simple polygon that contains all the points in the laser scan. Since the laser scan data is ordered by θ_i from $\theta_{\min}$ to $\theta_{\max}$, the bounding polygon can be constructed in one pass with the vertex sequence $\{\mathbf{c}\} \cup L \cup \{\mathbf{c}\}$. Fig. 1b shows an example of laser scan data and the corresponding bounding polygon.

2.2 Determining the Blind Regions

Using the laser scan data, we can determine which areas in the environment the car is unable to sense. We call these areas blind regions. The blind region, $\mathcal{B}$, is the set of points contained within a rectangle with a vertex sequence $\{L_i, L_i + k \cdot \hat{L}_{i,i+1}, L_{i+1} + k \cdot \hat{L}_{i,i+1}, L_{i+1}, L_i\}$ where $\hat{L}_{i,i+1}$ is the unit normal for the vector between points L_i and L_{i+1} that points away from the bounding polygon and k is a tuning parameter that contributes to the area of the blind region. In practice we only care for blind regions where $||L_i - L_{i+1}||_2 > \delta$ where δ is a tuning parameter because the laser scan consists of a finite number of points with a known angular distance. We will use $\mathcal{B}$ to denote the set of all such regions. Fig. 1c shows an example of blind regions in found in a found from a laser scan.

2.3 Computing the Exploratory Path

Using the blind regions, current configuration of the quadrotor, and the bounding polygon, we present an anytime algorithm that computes a collision free path for the quadrotor that maximizes the total observed area of the blind regions within a given time horizon. The algorithm builds a search tree starting from the current configuration of the quadrotor. It expands leaf nodes in descending order of total observed blind region area and only adds new leaf nodes to the search that are contained within the bounding polygon. When a collision free neighbour is propagated, the orientation, $\theta^*(x, \mathcal{B})$, that maximizes the area of the remaining blind region, $\mathcal{B}$, viewed at that configuration, x, is also added to the search tree. Below we formally define this orientation.

Definition 2. Let $\psi(x, \theta, \mathcal{B})$ be the set of points visible by the quadrotor at position $x \in \mathbb{R}^3$ with orientation θ. Let $\theta^*(x, \mathcal{B}) = \underset{0<\theta\leq 2\pi}{\arg\max}\ \psi(x, \theta, \mathcal{B})$. For convenience, we define $\psi^*(x, \mathcal{B}) = \psi(x, \theta^*(x, \mathcal{B}), \mathcal{B})$.

As the quadrotor follows the path, the planner is constantly replanning. To avoid oscillating between candidate paths, the quadrotor only follows a new path if its current path is no longer collision free or if the new path has a larger objective value.

Algorithm 2 Path planning for remote sensing UAV (looking around the corner)

Input:

- x_0: The initial position of the robot, $\mathcal{B}$: The blind region, $\mathcal{P}$: The bounding polygon

Output:

- $\Pi \subset \mathbb{R}^3 \times [0, 2\pi]$: A sequence of 3D positions and orientations representing the path

```
$Q \leftarrow \{(x_0, \theta^*(x_0, \mathcal{B}), \mathcal{B} \setminus \psi^*(x_0, \mathcal{B}))\}$
while $|Q| > 0$ do
    $(x, \theta, \mathcal{B}') \leftarrow \underset{\mathcal{B}' \in Q}{\arg\min}\ \textsc{Area}(\mathcal{B}')$
    if $\textsc{SearchTimeoutExpired}()$ then
        $\Pi \leftarrow \{\}$
        while $\textsc{HasParent}(x, \theta)$ do
            $\Pi \leftarrow \Pi \cup \{x\}$
            $(x, \theta) \leftarrow \textsc{Parent}(x, \theta)$
        return $\Pi$
    for all $x' \in \textsc{CollisionFreeNeighbours}(x, \mathcal{P})$ do
        $\theta' \leftarrow \theta^*(x', \mathcal{B}')$
        $Q \leftarrow Q \cup \{(x', \theta', \mathcal{B}' \setminus \psi^*(x', \mathcal{B}'))\}$
        $\textsc{Parent}(x', \theta') \leftarrow (x, \theta)$
    $Q \leftarrow Q \setminus \{(x, \theta', \mathcal{B}')\}$
return $\{\}$
```

At the start of Algo. 2, we initialize a priority queue that is used to store the leaf nodes of the search tree. Each node is comprised of the position of the quadrotor, $x \in \mathbb{R}^3$, the orientation of the quadrotor on the Z-axis, $\theta \in [0, 2\pi]$, and the remaining blind region, $\mathcal{B}$, that is left unobserved after the quadrotor reaches x with orientation θ. Until the search timeout has expired, collision free neighbours of x are added to the search along with their maximizing orientation and remaining unobserved blind regions. Once the search has expired, the path, Π comprised of 3D positions and orientations, that was able to view the largest cumulative blind region area starting from x_0 is returned. Fig. 1d shows an example of a path being computed to view the blind regions.

Since the obstacles and blind regions are two dimensional, we fix the altitude of the quadrotor in our experiments. In the future we would like to extend the blind region detection and planning to three dimensions by using a 3D point cloud sensor mounted on the ground vehicle.

2.4 Autonomous Landing

To enable autonomous landing, the quadrotor tries to maintain a static position relative to the car as it drives towards a parking spot. Once the vehicle has stopped, the quadrotor moves directly above the landing platform and proceeds to land on the platform.

3 Relative pose localization

From each UWB tag we receive a range measurement r_i in its own frame. From the quadrotor we receive velocity measurements $\boldsymbol{v}$, a yaw reading ψ_q, and an altitude measurement z_a in an ENU-aligned world frame. The frames and relative transforms of our system are visualized in Fig. 2. Given n UWBs, we define the measurement vector as $z = [r_1, \ldots, r_n, \boldsymbol{v}, z_a, \psi_q] \in \mathbb{R}^{n+5}$

Yaw orientation is calibrated at the start by lining up the quadrotor along the car's x-axis and measuring the yaw offset ψ_{off} between the car and the quadrotor. The yaw of the quadrotor is then given by $\psi = \psi_q - \psi_{\text{off}}$

One challenge we encountered in estimating the 3D position of the quadrotor was that the distance measurements from the UWBs showed larger errors when the UWB on the quadrotor was out of their plane. We therefore first estimate $\hat{\boldsymbol{p}}^{\text{odom}}$ using only use the quadrotor's onboard odometry readings $\boldsymbol{v}$ and z_a as the inputs to a UKF. We then use the estimated height, $\hat{\boldsymbol{p}}_{\boldsymbol{z}}^{\text{odom}}$ with the UWB range measurements r_i to estimate the quadrotor's x-y position, $\hat{\boldsymbol{p}}^{xy}$. We do this by first projecting each r_i onto the plane of the estimated height of the quadrotor:

$$r_i^{\text{proj}} = \sqrt{r_i^2 - (\hat{\boldsymbol{p}}_{\boldsymbol{z}}^{\text{odom}})^2}$$

We then find $\hat{\boldsymbol{p}}^{xy}$ by solving the nonlinear least squares optimization

$$h(\hat{\boldsymbol{p}}^{xy}) = \min_{\hat{\boldsymbol{p}}^{xy}} \sum_{i=1}^{n} ((r_i^{\text{proj}})^2 - \|\hat{\boldsymbol{p}}^{xy} - \boldsymbol{t}_{\boldsymbol{i}}^{xy}\|^2)^2$$

We then define the 3D position estimate to be $\hat{\boldsymbol{p}}^{ls} = [\hat{\boldsymbol{p}}^{xy}, \hat{z}]$. Next, we combine $\hat{\boldsymbol{p}}^{ls}$ and $\hat{\boldsymbol{p}}^{\text{odom}}$ in a second UKF to find a final position estimate $\hat{\boldsymbol{p}}$. Thus the final state estimate is $\hat{x} = [\hat{\boldsymbol{p}}, \psi]$.

4 Results

In this section we provide experimental results that validate our approach. A video accompanies this submission and is available at [21].

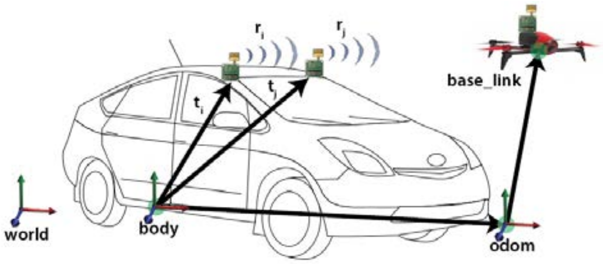


Fig. 2: The frames and measurements of our system.

4.1 Localization Accuracy

We tested our localization framework by emulating the car's UWB configuration inside a motion capture system. We placed motion capture markers on the quadrotor and on each UWB sensor. This allowed us to obtain the absolute position of the quadrotor and UWBs in the same coordinate frame. We then flew the quadrotor inside the motion capture system and recorded its predicted position determined by our localization and absolute position using the motion capture markers. We ran 10 tests and were able to obtain an error of 13.7 cm, or 35.9% the length of the quadrotor. Fig. 3 shows how our localization compares to the ground truth. The green and red lines respectively show the ground truth and predicted positions of the quadrotor. While our accuracy is less than the system in [10], the UWBs in that study were all in the same plane. The accuracy of out-of-plane position estimation using range measurements is lower than in-plane estimation because of the larger state space. Since we wanted our quadrotor to have the ability to fly beyond the plane of the roof-mounted UWBs, we included additional out-of-plane UWBs which decrease accuracy compared to having all UWBs in the same plane, but which help to provide greater accuracy for out-of-plane measurements.

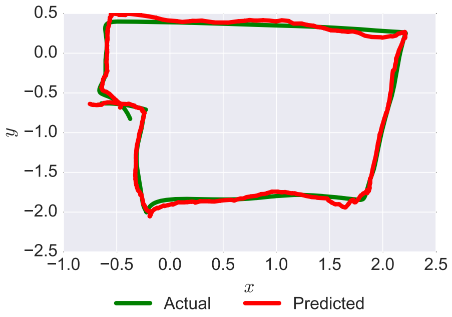


Fig. 3: Comparison of our localization method with respect to ground truth. Ground truth was supplied by a motion capture system.

4.2 Experimental Setup

For our experiments, we used a Toyota Prius with a SICK LMS1xx mounted on the front of the car and six Decawave TREK1000 UWBs mounted on the roof and front bumper of the car. A platform for the quadrotor to take off and land was attached to the front bumper of the Prius. We used a Parrot Bebop 2 quadrotor with a Decawave TREK1000 mounted on the battery. Fig. 4 shows the Toyota Prius and modified Bebop 2 quadrotor used in the experiments.

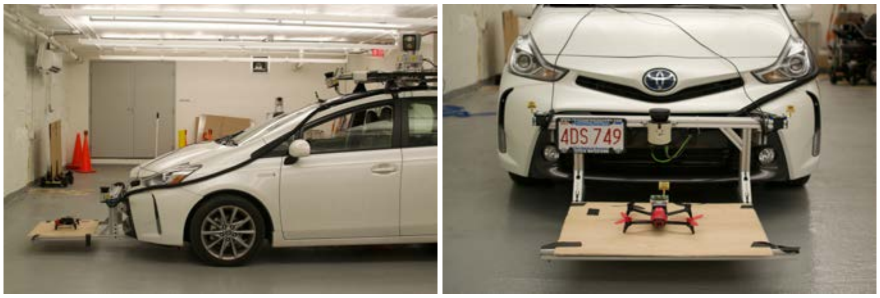


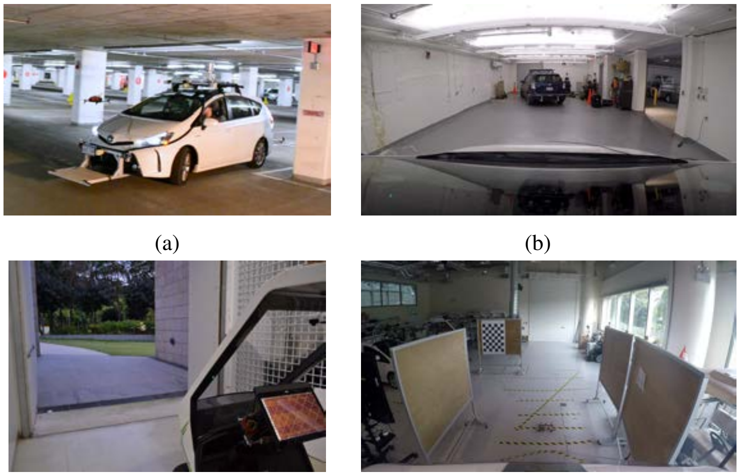
Fig. 4: Pictures showing Toyota Prius and Parrot Bebop 2 used in the experiments

We also ran tests using an autonomous golf cart as our ground vehicle in two different settings. One setting, shown in Fig. 5d, was artificially created using tall whiteboards as obstacles to mimic an adversarial environment. The second, shown in Fig. 5c, was a more realistic setting with the golf cart approaching an open garage door with blind spots on either side. In both cases, the quadrotor was successfully able to observe the blind spots and relay this information back to the computer on board the golf cart. For the interest of brevity, we will only discuss in detail the experiments using the Toyota Prius.

Our experimental scenario involves a car preparing to leave a garage with a significant blind spot. The car is unable to sense around the corner to determine if there are pedestrians or other cars that may obstruct its path. Our quadrotor takes off from the car's front bumper platform and autonomously flies out of the garage and looks around the corner. The car is then able to leave the garage when there are no more pedestrians detected by the quadrotor. Once the car is ready to return, it backs up into the garage. The quadrotor then follows the car into the garage and autonomously lands on the platform.

4.3 Experiment With Quadrotor

Under each experimental condition shown in Fig. 5, we conducted multiple tests. In the course of one afternoon we performed 25 tests in the environment shown



(a) (b)

(c) (d)

Fig. 5: Snapshots from four experimental settings in which we tested our algorithm. Figures (a) and (b) used a Toyota Prius in a parking lot and a garage, respectively. Figures (c) and (d) used an autonomous golf cart in outdoor and indoor environments.

in Fig. 5b in which the quadrotor successfully took off from the car, followed a path to observe blind spots, and landed back on the car's platform. Each test took around a minute to autonomously look around the corner and land back on the car. In every case, take off, path following, and landing was successfully completed. For the remainder of this section we will detail one representative experiment.

4.3.1 Looking Around the Corner

Fig. 6 shows snapshots of the experiment as it progressed. The first column is a third person angle of the Prius and the quadrotor. The second column shows frames from the quadrotor's on-board camera along with object detection and classifications from the convolutional neural net. The third column is a visualization of the sensor data from the car, the bounding polygon, blind regions, and the quadrotor's plan. Each row shows a single snapshot from the experiment.

The snapshots show that the quadrotor is able to successfully take off from the car, use the laser scan to find the blind regions, and plan a path to look around the corner in the garage. The last row shows that our system is able to detect the pedestrian around the corner and provide the bounding box back to the car.

Note that even though the quadrotor is not equipped with the sensors needed to perform robust 3D obstacle avoidance, it is able to avoid collisions and fly through the open garage door using the laser scan from the car.

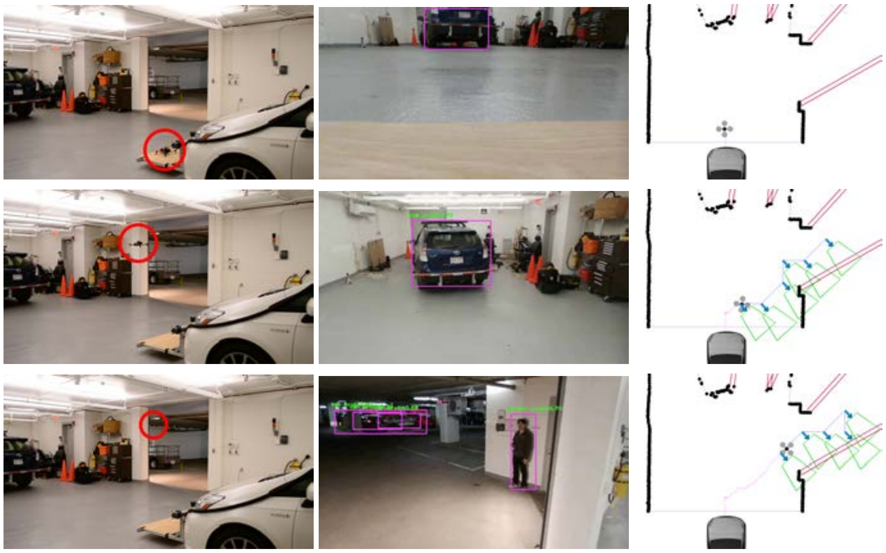


Fig. 6: Snapshots from the experiment as it progressed. The second column is from the quadrotor's on board camera. The third column shows a visualization of the planner.

4.3.2 Landing On the Car

Once the car is ready to park, the quadrotor is able to autonomously land back on the platform attached to the front bumper. Fig. 7 shows snapshots from the experiment as the quadrotor followed the car and landed on the platform. The first image in Fig. 7 shows the quadrotor following the car as it backs up into a garage. The second shows the car parked and the quadrotor hovering over the platform. The last image shows the quadrotor after it successfully landed on the platform.

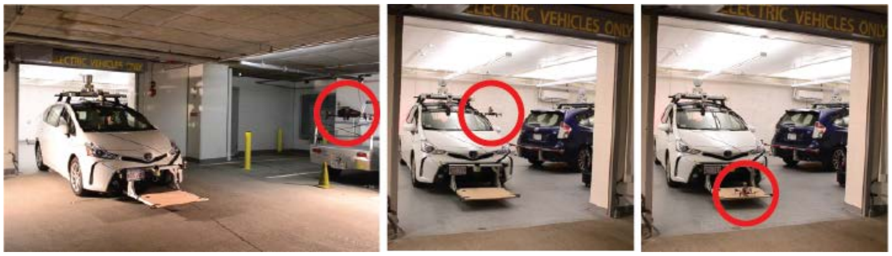


Fig. 7: Snapshots of the quadrotor landing on the car

5 Conclusion

In this work we presented a system for using a quadrotor to examine the blind spots of an autonomous car. We developed a path planning algorithm that maximizes visual coverage of blind spots in a 2D laser scan; created an experimental system using UWBs to localize the quadrotor with respect to the car; and performed tests in a variety of environments to verify the effectiveness of our system. Extensions to our work include planning using 3D laser scan data; improving the accuracy of the UWB localization; and applying our system to other problems such as package delivery and formation control. We believe that multi-robot coordination, particularly in the context of an autonomous car and a quadrotor, will become increasingly useful in the future.

References

[1] (2015) NHTSA traffic safety facts. https://crashstats.nhtsa.dot.gov/
[2] (2015) The Worlds first GPS MOOC and Worldwide Laboratory using Smartphones, ION Publications
[3] (2017) Decawave. `http://www.decawave.com/`, accessed: 2017-04-07
[4] ASIRT (2016) ASIRT association for safe international road travel 2016
[5] Borowczyk A, Nguyen DT, Nguyen APV, Nguyen DQ, Saussié D, Ny JL (2016) Autonomous landing of a multirotor micro air vehicle on a high velocity ground vehicle. arXiv preprint arXiv:161107329
[6] Bradley R (2016) Tesla autopilot
[7] Dagan E, Mano O, Stein GP, Shashua A (2004) Forward collision warning with a single camera. In: Intelligent Vehicles Symposium, 2004 IEEE, pp 37–42, DOI 10.1109/IVS.2004.1336352
[8] Delmerico J, Mueggler E, Nitsch J, Scaramuzza D (2017) Active autonomous aerial exploration for ground robot path planning. IEEE Robotics and Automation Letters 2(2):664–671
[9] Forster C, Pizzoli M, Scaramuzza D (2013) Air-ground localization and map augmentation using monocular dense reconstruction. In: IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), 2010, IEEE, pp 3971–3978
[10] Hepp B, Nägeli T, Hilliges O (2016) Omni-directional person tracking on a flying robot using occlusion-robust ultra-wideband signals. In: Intelligent Robots and Systems (IROS), 2016 IEEE/RSJ International Conference on, IEEE, pp 189–194
[11] Herissé B, Hamel T, Mahony R, Russotto FX (2012) Landing a vtol unmanned aerial vehicle on a moving platform using optical flow. IEEE Transactions on Robotics 28(1):77–89
[12] Hollinger GA, Djugash J, Singh S (2012) Target tracking without line of sight using range from radio. Autonomous Robots 32(1):1–14

[13] Kempke B, Pannuto P, Dutta P (2016) Harmonium: Asymmetric, bandstitched uwb for fast, accurate, and robust indoor localization. In: Proceedings of the 15th International Conference on Information Processing in Sensor Networks, IEEE Press, p 15

[14] Liu H, Darabi H, Banerjee P, Liu J (2007) Survey of wireless indoor positioning techniques and systems. IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews) 37(6):1067–1080

[15] Michael N, Shen S, Mohta K, Mulgaonkar Y, Kumar V, Nagatani K, Okada Y, Kiribayashi S, Otake K, Yoshida K, Ohno K, Takeuchi E, Tadokoro S (2012) Collaborative mapping of an earthquake-damaged building via ground and aerial robots. Journal of Field Robotics 29(5):832–841

[16] Mueller MW, Hamer M, D'Andrea R (2015) Fusing ultra-wideband range measurements with accelerometers and rate gyroscopes for quadrocopter state estimation. In: Robotics and Automation (ICRA), 2015 IEEE International Conference on, IEEE, pp 1730–1736

[17] Naegeli T, Alonso-Mora J, Domahidi A, Rus D, Hilliges O (2017) Real-time Motion Planning for Aerial Videography with Dynamic Obstacle Avoidance and Viewpoint Optimization. IEEE Robotics and Automation Letters PP(99)

[18] Nohmi M, Fujikura N, Ueda C, Toyota E (1978) Automatic braking or acceleration control system for a vehicle. US Patent 4,066,230

[19] Prorok A, Martinoli A (2014) Accurate indoor localization with ultra-wideband using spatial models and collaboration. The International Journal of Robotics Research 33(4):547–568

[20] Redmon J, Divvala S, Girshick R, Farhadi A (2016) You only look once: Unified, real-time object detection. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp 779–788

[21] Wallar A, Araki B, Chang R, Alonso-Mora J, Rus D (2015) Supporting video material for Foresight. `http://wallarelvo-tower.csail.mit.edu/fsr-foresight-video.mp4`

[22] Wenzel KE, Masselli A, Zell A (2011) Automatic take off, tracking and landing of a miniature uav on a moving carrier vehicle. Journal of intelligent & robotic systems 61(1):221–238