UAV Assisted Smart Parking Solution

Xin Li
CSE Department
Lehigh University
xil915@lehigh.edu

Mooi Choo Chuah
CSE Department
Lehigh University
chuah@cse.lehigh.edu

Subhrajit Bhattacharya
Mechanical Engineering Department
Lehigh University
sub216@lehigh.edu

***Abstract*—Smart parking solutions for big cities are critically important for reducing traffic congestion and vehicle energy consumption in big cities. Visual-based detection methods have been proposed since their maintenance costs are typically lower. However, visual based detection methods in existing systems are not robust due to varying light intensities, occlusions, and their deployment costs can still be expensive. In this paper, we present a collaborative UAV aided smart parking solution where a small team of low cost UAVs are used to collaboratively identify free parking spots of parking lots within campuses. We design a novel navigation control scheme to allow such UAVs to avoid obstacles and cover the parking lots sufficiently. In addition, we also present a visual detection scheme using the generative adversarial network (GAN) which allows us to predict parking availability without requiring pre-identification of parking spots. Simulation results indicate that our approach show promising results.**

I. Introduction

The advancement of microelectronics, wireless technologies has spurred the design and deployment of smart devices which affect many aspects of our daily lives. For example, smart watches can monitor their owners' physiological parameters; sensors monitoring air quality are deployed to monitor the levels of pollutants in big cities [1]; embedded devices in autonomous or semi-autonomous cars can recognize obstacles and apply the breaks to avoid accidents; smart thermostats can monitor home owners' preferences and automatically set temperature for appropriate room comfort. Smart devices have been deployed at a very fast rate due to their low costs and ease of deployment.

Smart cities can greatly benefit from this growing technological trend [2], [3] especially in the area of transportation, healthcare management, surveillance and city planning. Among these, parking management with smart technologies have received some attention [4], [3], [5]. Finding empty parking slot has become an everyday chore for many college students in campuses as well as for drivers in big cities. The conventional method of having drivers driving around the parking lots or streets to find a free spot is inefficient, time consuming and not environmental friendly. A study in 2007 [6] found that vehicles looking for parking over a year in a small business district of Los Angeles created the equivalent of 38 trips around the world, burning 47,0000 gallons of gasoline and producing 730 tons of carbon dioxide.

One key factor contributing to such excessive vehicle parking miles is a lack of real-time information about parking availability. Not all parking lots (especially those outdoor parking lots in campuses) have entry/exit counters installed to help one figure out the parking availability. Even if such counters are installed at the entrace/exit points in a multi-storey parking garage, we still need to know exactly which storey has empty parking spots. Different solutions for a parking guidance (PG) system have been proposed. One critical component of such a PG system is having the right detector for collecting parking availability information reliably and accurately and then present them in a user-friendly manner to drivers. The right detector should correctly report the parking availability information irrespective of environmental changes, e.g., day or night time, cloudy or sunny day, having different sizes of vehicles in the parking spots. The cost of deploying and maintaining such detectors in a PG system is also important. RFID based or ultrasound sensor based solutions typically are expensive since sensors need to be installed on each parking spot, and maintained.

Vision-based detectors are typically more cost efficient for each visual node (typically a camera and a transmitter) can monitor many vehicular spots simultaneously and the cost of maintenance is low. However, the deployment cost can still be high since cameras need to be installed and wireless connectivity needs to be provided to feed the collected videos to a remote server for image analysis. The reliability of a camera-based visual detector system is affected by varying light intensity, camera resolution and bad weather. Thus, it is important that researchers research on vision-based techniques that can improve the robustness of any visual parking guidance systems.

In this paper, we consider deploying a small team of low cost UAVs to perform visual inspection for parking availability. Our collaborative UAV aided smart parking solution utilizes one or a small team of UAVs equipped with appropriate cameras to survey large areas of open-space parking lots to determine parking availability with the help of a remote server. We design an efficient navigation control scheme to ensure that the team of UAVs can fly around potential obstacles, e.g., trees, and be able to cover different regions of a parking lot using their limited battery resources. In addition, our smart parking system utilizes a new generative adversarial network (GAN) model for detecting vacant and occupied parking spots. We show the robustness of our detection system by evaluating its performance using a large dataset of labeled parking spots. The evaluation results show promising results for our approach.

The rest of the paper is organized as follows: In Section II, we present some related work and background information about how GAN works. In Section III, we describe our system architecture and how our system works. In Section

IV, we describe our cooperative multi-UAV coverage control scheme. In Section V, we present the visual-based experimental results we obtained using the PKlot dataset. We conclude with discussions of near future work in Section VI.

II. Background & Related Work

A. Smart Parking Solution

As stated earlier, some work has been done in recent years to design smart parking solutions. Various parking vacancy solutions have been suggested e.g. in pavement wireless sensor networks [7], ultrasonic sensors [3], [8]. In [7], the authors design a sensor unit consisting of a GPS receiver and a passenger-side-facing ultrasonic rangefinder to identify urban street parking availability. Visual-based parking solutions have also been proposed [9], [10]. A recent work [11] has collected a large dataset of images from parking lots and applied a learning algorithm for free parking spots identification with acceptable results. In [12], the authors apply deep convolutional neural network method to identify parking vacancies. However, both papers assume that parking spots have been pre-identified from video frames and all one needs to do is to infer if there is a car in that parking spot. However, a more flexibile solution is to use a vision-based technique that can automatically identify parking spots and determine their availabilities.

B. Background of GAN

To automatically identify parking spots, in this paper, we use a Generative Adversarial Network (GAN) model recently proposed in [13]. A GAN is trained to identify the parking lots. The main idea behind a GAN is to have two competing neural network models. One model, called the generator, takes the original image as input and generates samples. The other model, called the discriminator, taking samples from the generator and the training data as its input and try to distinguish between these two sources (refer to Figure 1). During the training process, these two networks are trained simultaneously, and the generator is learning to generate more realistic samples while the discriminator is learning to perform better in distinguishing generated sample from training data.

We use the implementation of GAN released by Isola et al. [13]. They adapt the generator and discriminator architectures descripted in [14] in the following manners: 1. Typically, an encoder-decoder network used for many information translation problems allows low-level information to be shared between the input and output. Thus, it is desirable to shuttle this information directly across the net. To achieve this, the authors add skip connections following the general shape of a "U-Net" [15] (refer to Figure 2). 2. They also design a modified discriminator architecture, called patchGan, to model high-frequencies structure. This discriminator tries to classify if each $N \times N$ patch in an image is real or fake.

C. Collaborative UAV navigation control

Multi-robot coverage control is a well-studied problem. The traditional methods for attaining uniform coverage of an environment is the Lloyd's algorithm [16] and its continuous-time version due to Cortes [17]. While initially proposed for convex subsets of Euclidean spaces, in recent years significant advancements have been made to generalize the algorithms to general metric spaces which are possibly non-convex using graph search based methods [18], [19], [20]. Coverage control in the presence of dynamic weight functions have also been recently studied [21]. In this paper, we apply the fundamental graph-search based coverage control in on-convex environments along with a switching controller to make the UAVs visit different parking lots and attain visual coverage of them.

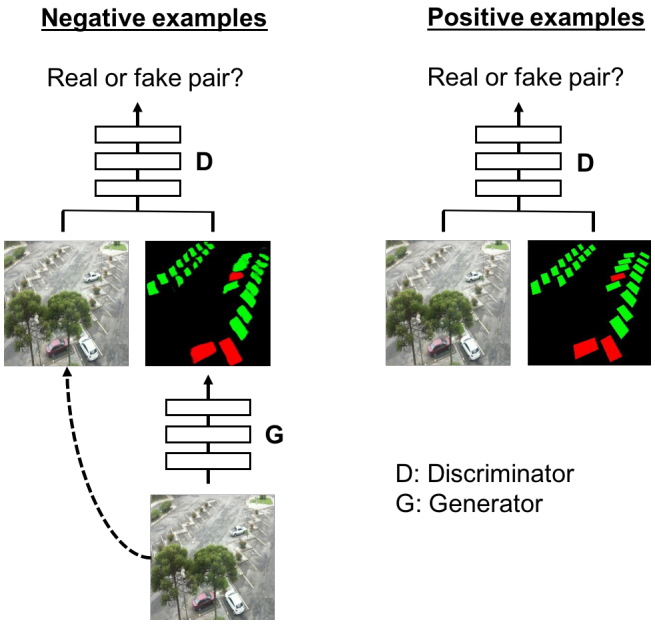


Fig. 1: Architecture of training a GAN to predict parking lots mask from original images. The discriminator, D, learns to classify between real and synthesized pairs. The generator, G, learns to fool the discriminator.

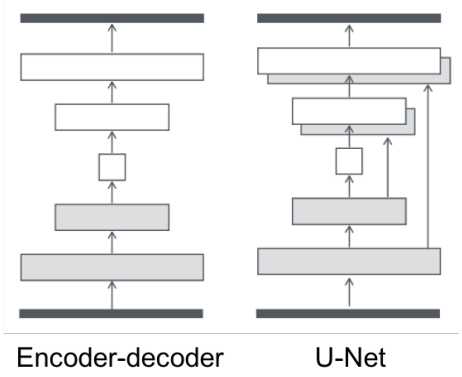


Fig. 2: Architecture of adapted generator. The "U-Net" is an encoder-decoder with skip connection between mirrored layers in the encoder and decoder stacks.

III. Proposed Architecture

In this section, we present an architecture to automatically detect parking vacancies in outdoor parking lots. Specifically, our scheme can detect the locations of parking spaces and predict if each space is vacant or not. Figure 3 shows the system architecture. Our scheme consists of three parts: **1. Drone:** One or more UAVs with a down-view camera will fly over the parking lots to take pictures and send them back to a remote server. **2. Server:** The server is used to receive pictures from the drone and automatically analyze parking lots on images and store the analyzed results in a database. The server is also responsible for controlling the navigations of cooperative UAVs such that these UAVs provide sufficient coverage of parking lots **3. Client:** This is an app running on a mobile device or a web browser. A user can submit his/her requests to the server to query the database regarding where the vacant parking spaces are right now. This architecture is similar to the one proposed in [12] by Valipour et al. Instead of using static cameras and assuming the locations of parking spaces are marked, we use a drone to collect pictures from parking lots which means the locations of parking lots are changing from image to image and there is no way to mark these locations manually. Thus, we employ a novel way to detect the locations of parking lots and predict the status of each parking space simultaneously.

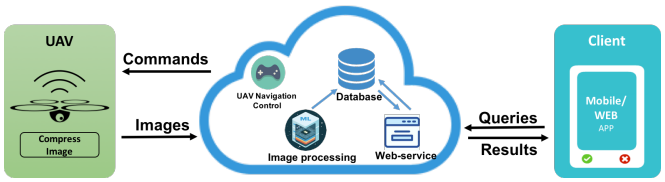


Fig. 3: System architecture.

A. *UAVs*

One or more UAVs are flown over the parking lots to collect images using their down-view cameras. After a UAV takes a picture with a high resolution, e.g. 1080*720, it first compresses the picture to a lower resolution, e.g. 256*256, and then sends the picture to a remote server via a wireless network. The image compression process runs fast and does not consume much resources (power and memory) of the computing unit within a UAV. Such image compression helps to reduce the communication cost. Instead of sending all frames, the UAV sends a frame to the server at a pre-designed frequency, e.g. a frame per second. The reason is twofold: (1). Usually, there is no large change in parking availability within a few seconds. (2). Lowering the sending frequency helps to save communication and power cost. In our system, the UAV does not need to save any image. Any UAV with a down-view (or front-view) camera and wireless communication capability can be used in our system. In addition, considering that the image will be compressed before sending to the server, the resolution of the camera on the UAV can be low, e.g., 256*256. As a result of these, we can use a normal and cheap UAV.

B. *Server*

The server in our scheme has four responsibilities: (1) UAV navigation control to ensure complete coverage of any parking lot surveyed; (2) Receiving images from UAVs; (3) Analyzing the images to locate the parking spaces and predicts their status (vacant or occupied); (3)Hosting a database to store the image processing results and some related information (time). (4) Providing an interface to receive queries from users and sends the results back to the users.

- **UAV navigation control:** The server will communicate with the deployed UAVs to provide navigation control information such that the UAVs can collaboratively capture images that cover the whole parking lot that they are supposed to survey;
- **Receiving images:** A UAV is connected to the server via WiFi or cellular data connectivity. The sever listens to a specific port to receive the compressed images from the UAVs. After receiving an image, the server passes it to the image processing model to analyze the image.
- **Image processing:** We trained a GAN (Generative Adversarial Network) model [13] to process the images. The input of the model is a compressed image from the drone with a resolution of 256*256 and the output of the model is an RGB image (same resolution as the input) which marks each parking space with different colors based on their statuses (e.g. green color indicates vacant while read color indicates occupied). Please refer to Section III-D for more implementation details. After analyzing, the server stores the status of each parking space and its associated time in the database.
- **Database:** the server hosts a database to store the parking spot availability information obtained from the image analyzer module. In our scheme, we only store some simple information, e.g. the status of each parking lot and the last-update time, in the database. Considering the older records are useless for users, we overwrite the previous status of a parking space if its status has changed.
- **Web-service interface:** This web-service interface provides a way for users to access the data stored in the database. A user submits a query to the server using the mobile or web app to check if there is any vacant parking space and the corresponding location. After receiving a query from the user, the web-service searches among records in the database and sends the results back to the user.

C. *Client*

The user client is a mobile device app or a web app. This app provides a way for a user to submit a query to the server to get the current status of the parking lots. After receiving the response from the server, the app displays the results to the user so that the user can easily know the location of those vacant parking spots.

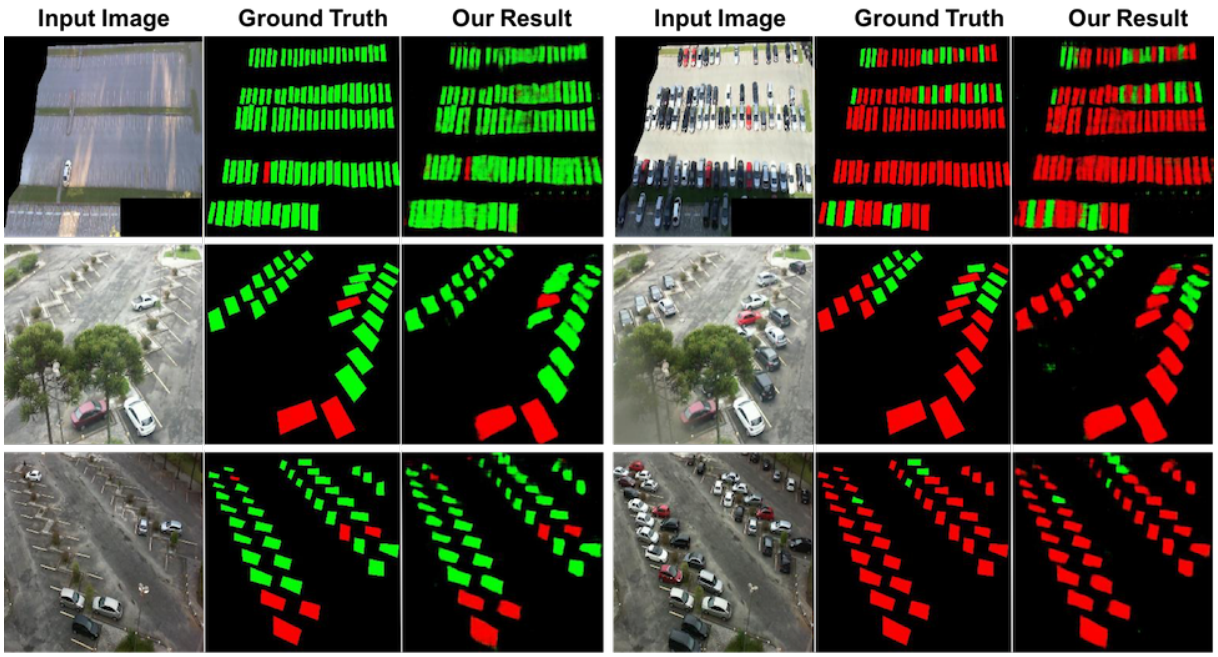


Fig. 4: Output samples of our model.

D. *Implementation Details*

In this subsection, we report the details of our system implementation.

- **UAV:** A Parrot AR. Drone 2.0 with a 720p camera is used. We use a Raspberry Pi 3 Model B to perform image compression algorithm which is implemented using Python programming language and Opencv library [22]. The image is compressed from the resolution of 1280*720 to 256*256. The UAV is connected to the server via a WiFi network.
- **Server:** We use a laptop with a 2.5GHz intel Core i7, 16GB Memory, and running MacOS Sierra as the server. **Image receiver:** We implement a socket server using Python programming language to listen and receive images from any deployed UAV. **Image analyzer:** In our system, a GAN model is trained to predict the parking lots. The generated sample is a mask which indicates the locations and labels of parking spaces. During our training process, we feed the original images and their corresponding ground truth to the model to train the generator and discriminator. In order to avoid overfitting and expand training data, we edit the training images by reversing, translating, and zooming operations. During testing process, we only feed the original image into our model and use the generator to generate the sample (mask) for us without using the discriminator. **Web-service interface:** The web-service interface is implemented using PHP programming language and published using the Apache Server.
- **Client:** We run an Android app on a Samsung Galaxy S5 with 2GB RAM. A user can submit his query to the sever and receive the response from the server using this Android App. Figure 5 shows the screenshot of our user interface.

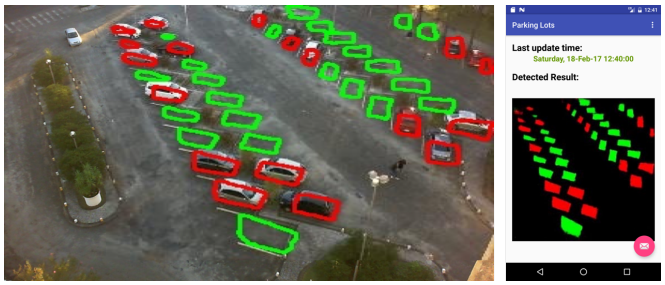


Fig. 5: The detected result shown on mobile device.

IV. Cooperative Multi-UAV Coverage Control

In order for the UAVs to effectively collect data on the availability of the parking spots, they need to coordinate and attain an uniform coverage of the parking lot(s) in a region of interest. To this end we use a variant of the continuous-time version of Lloyed's algorithm [16] first proposed by Cortes [17]. The algorithm, originally proposed for convex subsets of the Euclidean plane, is based on minimization of a *coverage functional* – a measure of how poorly an environment is covered.

A. *Background*

Suppose n UAVs want to attain coverage of an environment, Ω. The position of the k^{th} UAV is represented by $\mathbf{p}_k \in \Omega$. One also defines a weight function, $w : \Omega \to (0, 1]$, which indicate the importance of the different parts of

the environment – higher indicate greater importance. The *Voronoi Partition* of the environment is then defined as

$$V_k = \{\mathbf{q} \in \Omega \mid \|\mathbf{q} - \mathbf{p}_k\| \leq \|\mathbf{q} - \mathbf{p}_l\|, \forall l \neq k\} \quad (1)$$

V_k is thus he set of points that are closer to the k^{th} UAV than any other UAV in the environment. This in turn is used to define the Coverage Functional

$$\mathcal{H}(P) = \sum_{k=1}^{n} \int_{V_k} \|\mathbf{q} - \mathbf{p}_k\|^2 \; w(\mathbf{q}) \; \mathrm{d}\mathbf{q} \quad (2)$$

where $P = \{\mathbf{p}_1, \mathbf{p}_2, \cdots, \mathbf{p}_n\}$ and $\mathrm{d}\mathbf{q}$ is the differential volume element.

The name "*Coverage Functional*" is indicative of the fact that $\mathcal{H}$ measures how *bad* the coverage is — *i.e.*, the more well-distributed the UAVs are throughout the environment, the lower is the value of $\mathcal{H}$. This leads us to a control law for the UAVs that would minimize $\mathcal{H}$ – *i.e.*, to follow the negative of the gradient of $\mathcal{H}$. That is,

$$\dot{\mathbf{p}}_k = -\alpha \frac{\partial \mathcal{H}}{\partial \mathbf{p}_k} = 2A_k(\mathbf{p}_k - \mathbf{p}_k^*) \quad (3)$$

where, $A_k = \int_{V_k} w(q) \; \mathrm{d}\mathbf{q}$ is the *area* of the Voronoi cell, V_k, and $\mathbf{p}_k^* = \frac{1}{A_k} \int_{V_k} \mathbf{q} \; w(\mathbf{q}) \; \mathrm{d}\mathbf{q}$. Since the Voronoi cells, V_k, depend on the positions of the UAVs, P, the last equality in (3) requires the methods of differentiation under integration to be derived [17], [23].

B. Discrete Graph-based Method for Non-convex Environments

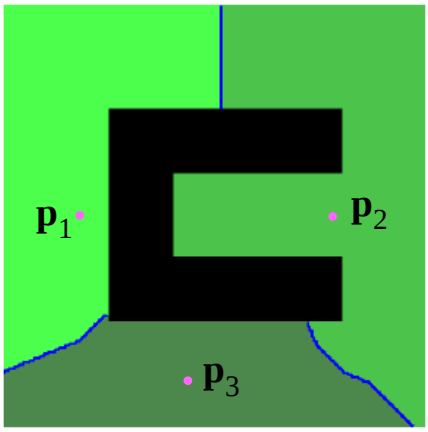


Fig. 6: Voronoi partition of a non-convex subset of the Euclidean plane with 3 UAVs computed using a discrete graph search based wavefront algorithm.

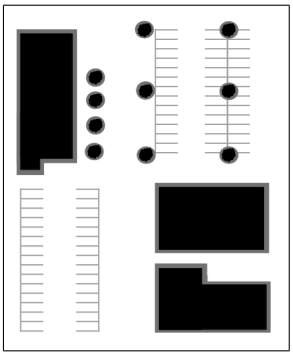
The problem with the above control algorithm is that it is valid only when Ω is a convex subset of an Euclidean space. However, in [18] it was shown that it is possible to extend this control algorithm to more general metric spaces which are possibly non-convex with obstacles. The key to that extension lies in observing that the 2-norm, $\|\cdot\|$ in (1) and (2) needs to be replaced by the appropriate and more general distance function $d(\cdot, \cdot)$ in a general (non-Euclidean and non-convex) metric space Ω. In particular, if Ω is a general non-convex subset of an Euclidean space (more generally, a space with *isotropic Riemannian metric*), and $d(\mathbf{u}, \mathbf{v})$ the *geodesic distance* (length of shortest path) between points $\mathbf{u}, \mathbf{v} \in \Omega$, then the general control law reduces to

$$\dot{\mathbf{p}}_k = -2\alpha \int_{V_k} d(\mathbf{q}, \mathbf{p}_k) \; \mathbf{z}_{\mathbf{q}\mathbf{p}_k} \; w(\mathbf{q}) \; \mathrm{d}\mathbf{q} \quad (4)$$

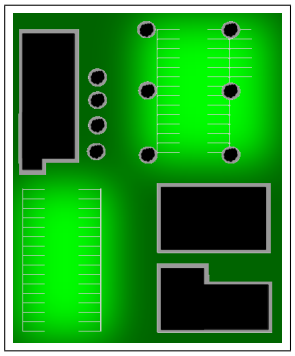
where $\mathbf{z}_{\mathbf{q}\mathbf{p}_k}$ is an unit tangent at $\mathbf{p}_k$ to the geodesic (shortest path) connecting $\mathbf{p}_k$ and $\mathbf{q}$.

Using graph search-based wavefront propagation method, both the generalized Voronoi partition (Figure 6) and the control law in (4) can be computed very efficiently for all the UAVs [18] and hence the generalized coverage functional in the non-convex space can be minimized to attain a *centroidal Voronoi tessellation*. In addition, to ensure that the tessellation boundary of the UAVs not just split area equally, but split the total weights (weighted area) relatively equally, we use a distance function, d, that is weighted by the w as well (this results in a non-uniform, but isotropic metric). A graph search based method allows us to implement such a distance function simply by weighing the edges of the graph by w. The details of the method can be found in [18], [20].

C. Weight Function and Metric for Parking Lot Coverage



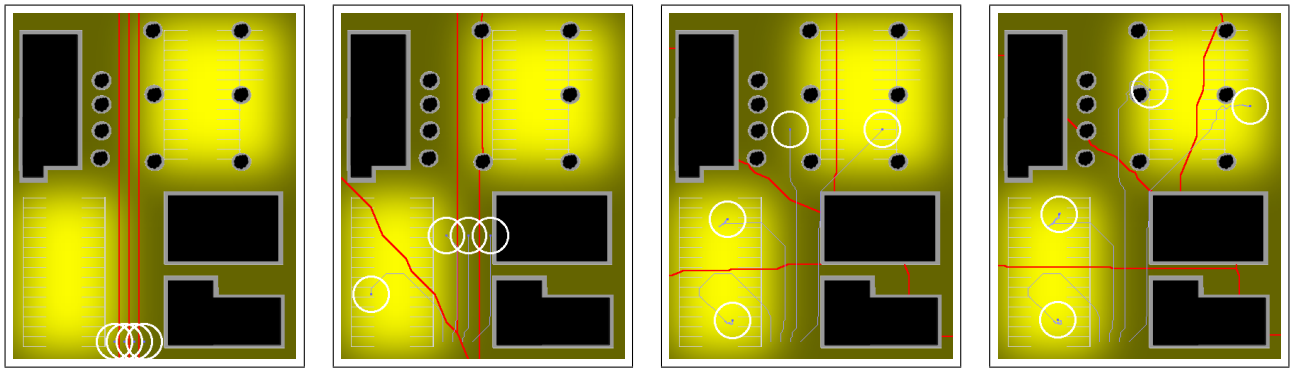
(a) An environment with obstacles (buildings and trees) and parking lots.



(b) The weight function (indicated by intensity of green) giving higher priority to the parking lots.

Fig. 7: Illustration of the construction of the weight function to prioritize parking lots.

The role of the weight function, w, is to prioritize parts of the environment, Ω, that need to be covered/surveyed more than the other regions of the environment. In the parking lot monitoring application, we thus exploit the weight function to prioritize the regions of the accessible (obstacle-free) environment that constitute the parking lot. In practice it is possible to automate the identification of parking lots in a map of the environment in terms of bounding boxes. However, since the UAVs' environment (location of parking lots and obstacles) itself does not change, we manually construct the bounding boxes for the parking lots, set high (~ 1.0) weights for regions inside the bounding boxes (and



(a) Robots start at the bottom of the environment at $t = 0$.

(b) $t = 150$.

(c) $t = 300$.

(d) $t = 400$ (convergence attained).

Fig. 8: Four UAVs (highlighted by white circle around them) attain coverage in an environment by following the coverage functional minimizing control law of (4). The intensity of yellow indicates the weight function. The red curves indicate the boundary of Voronoi cells at every iteration. At $t = 0$ the UAVs start at the bottom of the environment. By $t = 400$ the UAVs have attained a centroidal Voronoi tessellation.

a low weight, ϵ, for the other regions), and apply a Gaussian blur to construct a smooth weight function (Figure 7).

UAVs starting at a base station and following the control law (4) will navigate the environment avoiding obstacles, attain coverage of the environment, while prioritizing the high weight regions (parking lots). This is illustrated in the simulation of Figure 8. Four UAVs start at the bottom of an environment, and eventually attain coverage of the two parking lots in the environment, with two UAVs stationed at each parking lot.

D. Sequential Coverage of Multiple Parking Lots Using Switching Controller

When there are multiple parking lots to be covered, and not enough UAVs available to simultaneously cover all, we use a switching controller. In each of the discrete state of the controller, the UAVs are tasked with covering one parking lot (the assignment being enforced by setting high weight over the particular parking lot), and we switch from one parking lot to another once centroidal coverage of the previous lot is attained (*i.e.* convergence is reached). This also lets us cycle between parking lots indefinitely in order to maintain persistent coverage of the environment. For travel between parking lots (the intermediate state between the switch from one parking lot to another), in order to increase speed and efficiency, the UAVs use shortest paths. The simulation in Figure 9 shows two UAVs covering two parking lots in the Lehigh university campus in a sequential manner.

V. Vision-Based Experimental Results

In this section, we report the experimental results of our vision-based solution using a well known parking lot dataset, namely PKLot dataset [11]. PKLot contains 12,417 images with a resolution of 2180*720 captured from two different parking lots in sunny, cloudy and rainy days. The first parking lot has two cameras with opposite capture angles, namely UFPR04 and UFPR05. The another parking lot is named PUCPR. Each image of the dataset has a corresponding XML file including the coordinates of all the parking spaces on this image and their labels (occupied/vacant).

A. Training Data Generation

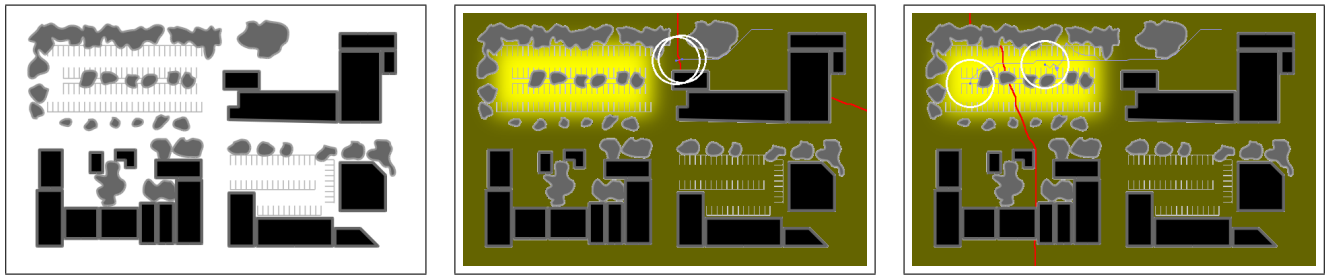
In order to generate the ground truth to train our model, we read the coordinates and label of each parking space from the XML files and mark its region on a black panel with different colors, e.g. red color indicates the parking space is occupied and green color indicates the parking space is vacant (refer "Ground Truth" columns in Figure 4). During our experiment, we found there are some images (images in folder PUCPR/Sunny/2012-11-07, PUCPR/Sunny/2012-11-06, and PUCPR/Sunny/2012-10-30) which are falsely labeled, so we remove these images from the dataset. In addition, considering that no all parking spaces on the images in subset PUCPR are labeled, we use black masks to cover those areas without any label. Eventually, we have 12,162 images in total and we use 50% images for training and another 50% for testing (the same procedure of the authors in [11]). Table I shows the details of our training and testing sets.

B. Results

In order to evaluate the performance of our system, two sets of experiments are conducted:

- **Multiple parking lots training and testing:** The model is trained on multiple training subsets, and then tested on all testing subsets.
- **Multiple parking lots training and testing on rotated images:** To emulate the fact that images collected by a UAV may have varying orientations, we create a synthetic testing dataset by rotating the existing images by a small angle (within 10 degrees). Then, we use a trained model created using the existing training subset, and test the model using test images rotated with some random angles.

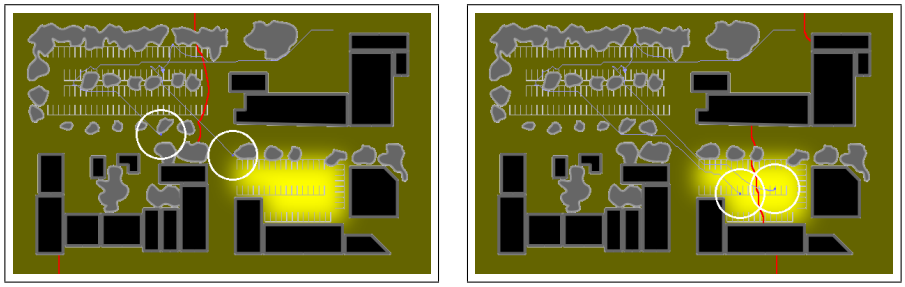
All reported results in this section are collected using a trained model generated with 50 training epochs.



(a) Two parking lots near the Mohler lab in Lehigh University campus (buildings in black, trees in dark gray).

(b) Two UAVs fly towards the first parking lot after starting at the upper right corner of the environment.

(c) They attain centroldal Voronoi coverage of first parking lot.



(d) The UAVs fly towards the second parking lot.

(e) And attain centroldal Voronoi coverage of that.

Fig. 9: Switching control for covering two parking lots using 2 UAVs. (a),(b): In the first state the UAVs attain overage of the first lot since the weight id high on that lot. (c)-(d): In the second state the weight shifts to the other parking lot and the UAVs attain coverage of that.

TABLE I: Training and testing sets

		Training Sets	Testing Sets
PUCPR	Sunny	1029	1032
	Cloudy	664	664
	Rainy	416	415
	Total	**2109**	**2111**
UFPR04	Sunny	1050	1048
	Cloudy	704	704
	Rainy	143	142
	Total	**1897**	**1894**
UFPR05	Sunny	1250	1250
	Cloudy	713	713
	Rainy	113	113
	Total	**2076**	**2076**
In total		**6082**	**6081**

either during a cloudy day or a sunny day from each subset of the PKLot images (1st row input images are from PUCPR subset, 2nd row and 3rd row are from UFPR04 and UFPR05 respectively), what ground truth parking availability looks like for each input image, and what our predicted output shows.

TABLE II: Precision of training on all 3 training subsets

		Trained on all 3 subsets		
		Occupied	Vacant	Overall
Tested on (50 epochs)	UFPR04	98.21%	97.50%	97.81%
	UFPR05	99.1%	98.0%	98.6%
	PUCPR	60.4%	71.9%	67.0%
	Overall	**77.4%**	**80.7%**	**79.2%**
Tested on (200 epochs)	UFPR04	98.3%	96.8%	97.5%
	UFPR05	96.8%	93.8%	95.6%
	PUCPR	94.5%	94.9%	94.7%
	Overall	**95.7%**	**95.0%**	**95.3%**

1. Multiple parking lots training and testing

In Table II, we report the precision of our model trained using the combined three training subsets, and tested individually using each test subset. It is obvious that our model generates a good result in different viewpoints and weather conditions. From this result, we notice that the accuracy of predicting a spot is occupied is higher than predicting a spot is vacant. The reason is that a vacant parking space has similar feature or texture as the ground and hence harder to distinguish, while an occupied parking space is different from the ground. Figure 4 shows some samples of the output of our model. In each row, we show an input image taken

2. Multiple parking lots training and testing on rotated images In this experiment, we rotated the testing images by a random angle (within a range of (-10, 10)). Table III shows the precision of our trained model for this task. From the result, we see that the model maintains a high precision on the UFPR04 and UFPR05 testing subsets, but performs worse on the PUCPR testing subset. The reason is twofold: (1) the parking spaces on the images of the PUCPR subset are not very clear compared to the other two subsets, (2) the trained model is more sensitive to the orientation changes of the rotated PUCPR test images because the parking spots within the PUCPR images are packed more closely. To rectify such problem, we will design the navigation control of our UAVs

such that they take images at a closer range as long as their flight paths are not hampered by obstacles.

TABLE III: Precision of training on multiple parking lots training and testing on rotated images

		Trained on all 3 subsets		
		Occupied	Vacant	Overall
Tested on	UFPR04	93.0%	87.0%	89.6%
	UFPR05	94.6%	93.2%	94.0%
	PUCPR	39.7%	66.2%	56.0%

VI. Conclusion

In this paper, we proposed a UAV-assisted architecture to detect the status (occupied/vacant) of parking spaces in parking lots. We use a novel generative model, GAN (Generative Adversarial Network), to automatically detect the locations of parking spaces and predict their occupancy states. The performance of our vision-based scheme is evaluated using a well known PKLot dataset. The result shows that our scheme achieves a high detection and prediction result. In addition, we also propose a novel algorithm to control the navigation of these UAVs so that they can collaboratively cover the whole parking lot using limited battery resources that they have and also avoid obstacles such as trees, occlusions while flying. In the near future, we hope to investigate the robustness of our design using more challenging parking lot and side street parking images that contain obstacles e.g. trees, occlusions along the flight paths of collaborative UAVs.

References

[1] Y. Zheng, X. Yi, M. Li, R. Li, Z. Shan, E. Chang, and T. Li, "Forecasting fine-grained air quaity based on big data," *Proceedings of ACM KDD*, 2015.

[2] A. Zannella, N. Bui, A. Castellani, L. Vangelista, and M. Zorzi, "Internet of things for smart cities," *IEEE Internet of Things Journal*, 2014.

[3] S. Mathur, T. Jin, N. Kasturirangan, J. Chandrashekharan, W. Xue, M. Gruteser, and W. Trappe, "Parknet: Drive-by sensing of road-side parking statistics," *Proceedings of ACM MobySys*, 2010.

[4] R. Lu, X. Lin, H. Zhu, and X. S. Shen, "Spark: A new vanet-based smart parking scheme for large parking lots," *Proceedings of IEEE INFOCOM*, 2009.

[5] Y. Geng and C. Cassandras, "New smart parking system based on resource allocation and reservations," *IEEE Transactions on Intelligent Transportation Systems*, vol. 14, pp. 1129–1139, 2013.

[6] D. Shoup, "Cruising for parking," *Access*, vol. 30, pp. 16–22, 2007.

[7] R. Bajwa, R. Rajagopoal, P. Varaiya, and R. Kavaler, "In-pavement wireless sensor network for vehicle classification," *Proceedings of ACM Information Processing in Sensor Neworks (IPSN)*, 2011.

[8] A. Kianpisheh, N. Mustaffa, P. Limtrairut, and P. Keikhosrokiani, "Smart parking system architecture using ultrasonic sensors," *International Journal of Software Engineering and Its Applciations*, 2012.

[9] C. C. Huang and S. J. Wang, "A hierarchical bayesian generation framework for vacant parking space detection," *IEEE Transactions on Circuits and Systems For Video Technolog*, 2010.

[10] H. Ichihashi, T. Katada, M. Fuijiyoshi, A. Notsu, and K. Honda, "Improvement in the performance of camera based vehicle detection for parking lot," *IEEE International Conference on Fuzzy Systems (FUZZ)*, 2010.

[11] P. R. De Almeida, L. S. Oliveira, A. S. Britto, E. J. Silva, and A. L. Koerich, "Pklot–a robust dataset for parking lot classification," *Expert Systems with Applications*, vol. 42, no. 11, pp. 4937–4949, 2015.

[12] S. Valipour, M. Siam, E. Stroulia, and M. Jagersand, "Parking stall vacancy indicator system based on deep convolutional neural networks," *arXiv preprint arXiv:1606.09367*, 2016.

[13] P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros, "Image-to-image translation with conditional adversarial networks," *arXiv preprint arXiv:1611.07004*, 2016.

[14] A. Radford, L. Metz, and S. Chintala, "Unsupervised representation learning with deep convolutional generative adversarial networks," *arXiv preprint arXiv:1511.06434*, 2015.

[15] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," in *International Conference on Medical Image Computing and Computer-Assisted Intervention*. Springer, 2015, pp. 234–241.

[16] S. Lloyd, "Least squares quantization in PCM," *IEEE Trans. Inform. Theory*, vol. 28, no. 2, pp. 129–137, 1982.

[17] J. Cortes, S. Martinez, T. Karatas, and F. Bullo, "Coverage control for mobile sensing networks," *IEEE Trans. Robot. Autom.*, vol. 20, no. 2, pp. 243–255, Apr. 2004.

[18] S. Bhattacharya, R. Ghrist, and V. Kumar, "Multi-robot coverage and exploration on riemannian manifolds with boundary," *International Journal of Robotics Research*, vol. 33, no. 1, pp. 113–137, January 2014, dOI: 10.1177/0278364913507324.

[19] J. Durham, R. Carli, P. Frasca, and F. Bullo, "Discrete partitioning and coverage control for gossiping robots," *IEEE Transactions on Robotics*, vol. 28, no. 2, pp. 364–378, 2012.

[20] S. Bhattacharya, N. Michael, and V. Kumar, "Distributed coverage and exploration in unknown non-convex environments," in *Proceedings of 10th International Symposium on Distributed Autonomous Robotics Systems*. Springer, 1-3 Nov 2010.

[21] J. M. Palacios-Gass, E. Montijano, C. Sags, and S. Llorente, "Distributed coverage estimation and control for multirobot persistent tasks," *IEEE Transactions on Robotics*, vol. 32, no. 6, pp. 1444–1460, Dec 2016.

[22] *The OpenCV Reference Manual*, 2nd ed., Itseez, April 2014.

[23] L. C. A. Pimenta, V. Kumar, R. C. Mesquita, and G. A. S. Pereira, "Sensing and coverage for a network of heterogeneous robots," in *Proc. of the IEEE Conf. on Decision and Control*, Cancun, Mexico, Dec. 2008, pp. 3947–3952.