Perceiving Signs for Navigation Guidance in Spaces Designed for Humans

Claire Liang*, Cheng Perng Phoo*, Laasya Renganathan†, Yingying Yu*, Bharath Hariharan*, and Hadas Kress-Gazit‡
*Department of Computer Science
†Department of Electrical Engineering
‡Sibley School of Mechanical and Aerospace Engineering
Cornell University, Ithaca, New York 14853
Email: cliang@cs.cornell.edu, cpphoo@cs.cornell.edu, lpr46@cornell.edu, yy759@cornell.edu,
bharathh@cs.cornell.edu, hadaskg@cornell.edu

***Abstract*—Robot navigation in spaces like airports, malls and stadiums relies on provided, high-detail, metric maps. The space is pre-programmed into the system and the robot does not use context and semantic information about their surroundings to inform navigation. As a consequence, these deployed robot systems can be brittle to inconsistencies between the dynamic world around them and the perfect map they plan in. We work towards a robot system that uses semantic information in the form of available signage — as designed for humans — in these environments designed for humans. In this work, we use a formal abstraction of "signage" to ground our robot perception system, intended for a low-cost, simple monocular camera input robot. We present the "signage" representation, the architecture of this perception system, initial results from our perception system, and perform simulated navigation using interpreted signs using real airport maps.**

I. Introduction

Guide robots in spaces like airports, malls, and office buildings are becoming a new normal. However, they are often seen as fun toys rather than crucial and efficient tools for guidance and navigation. Their interactions with humans can often be awkward and in some instances, they fail entirely to aid a human with their navigation task.

Most deployed real-world robots are dependent on metric maps and plan using metrically defined trajectories. Humans, on the other hand, often communicate spatial directions to their peers in terms of identifiable landmarks.

Consider the airport sign shown in Fig. 1. Human pedestrians in these environments are able to use these signs to sequence together a plan to their final destinations. Robots deployed in these spaces are given annotated, detailed maps, and are unable to dynamically adapt to unplanned closures or detours in their space. Our goal is to create a robot system that identifies, interprets, and uses these signs to effectively plan in spaces designed for humans, given no map.

In this workshop paper we focus on building and evaluating the sign recognition portion of our full robot system. We begin by introducing the problem formulation, the robot and its sensing representation, as well an abstraction for signs. We then describe the design of our vision-based perception system to detect and interpret signs into our proposed abstraction. Then, we present initial results from each stage of our perception system, as well as how the results from our perception system impact simulated navigation in real airport maps.

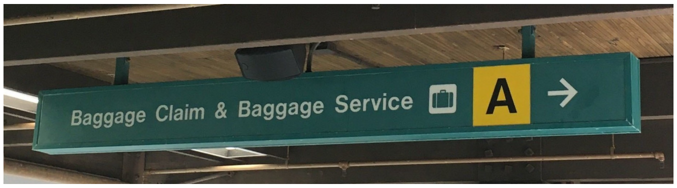

Figure 1. An example sign in an airport. The sign contains semantic content "Baggage claim & Baggage Service" as well as an arrow conveying high level directions.

II. Related Work

Methods from the area of semantic mapping often jointly estimate hybrid metric, topological, and semantic representations of an environment and use this representation to navigate. However, these techniques also often require a hand annotated topological graph representation of landmarks within a scene [22, 5], and thus do not scale well for use across a large range of environments. They are also less applicable to environments that have rapidly changing configurations or less distinct landmarks [7, 23]. Furthermore, semantic mapping and planning rarely leverages the context and clues that landmarks and the environment around them provides.

In order to effectively use signs for navigation, the robot's perception system must be able to both detect and recognize signs. Real-world signage presents a series of complicating factors: sign text can vary in font shape or style, have any text orientation, and be present on any background color or shape. While sign detection is not an entirely novel problem, gathering the semantic information from sign content itself is much less explored. Existing work in the domain of sign detection often addresses specific sets of signs, such as road or traffic signs, and intentionally exploits road sign structure (such as color and shape). For example, [2, 21] use color segmentation and shape-based template matching to detect traffic signs and [4] cast sign detection as classification from shapes or symbols. These methods cannot generalize to general, unstructured signs at the level necessary for robust navigation.

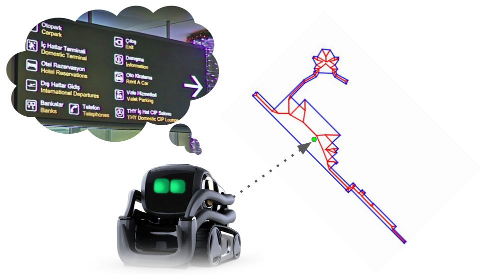

Figure 2. An illustration of how a real sign is used by the robot when navigating in a space. The robot uses the sign to select which edges to travel on in a space skeletonization structure.

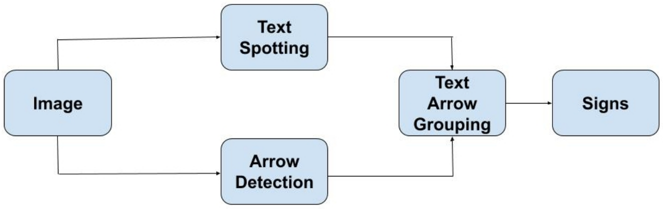

Figure 3. Structure of how signs are detected and interpreted using our vision system

Our perception system builds on general object detection [10, 9, 20, 11, 14, 19, 18, 15, 1]. Airport sign data is expensive to collect and methods that develop sign-specific detectors often require more training data than we can feasibly gather. Therefore, recent work in text spotting[1] [3, 12, 16, 17, 25], can bolster our perception system. We develop two separate detectors - one for the text and another for the arrow on the sign. Specifically in this iteration of our system, we build upon the current state-of-the-art text spotter - CharNet [25] for text spotting and Mask-RCNN with Feature Pyramid Network ([11, 14]) for arrow detection.

III. Methods

A. Sign Representation

Assume a closed, bounded, polygonal freespace P with a skeletonization $\Gamma(P)$. The assumptions and method used to generate our skeletonization is addressed in prior work [13]. We assume the skeletonization can be represented as a graph G with vertices V and edges E. For our implementation, we choose to use the Medial Axis (MA) as our skeletonization [6]. For a given goal point g, each sign s in the environment consists of two pieces of data: the position of the sign (x, y) and the information (σ) on the sign. The σ representation can vary depending on perception and sensing capabilities. For this work, the robot must be able to identify any arrows on a sign along with their associated text and convert the sign's arrow to a valid heading angle for navigation. We describe σ in the next section.

B. Vision System Structure

Our vision system takes in an image and identifies signs. Then, produces the σ for each sign. Each sign's σ consists of:

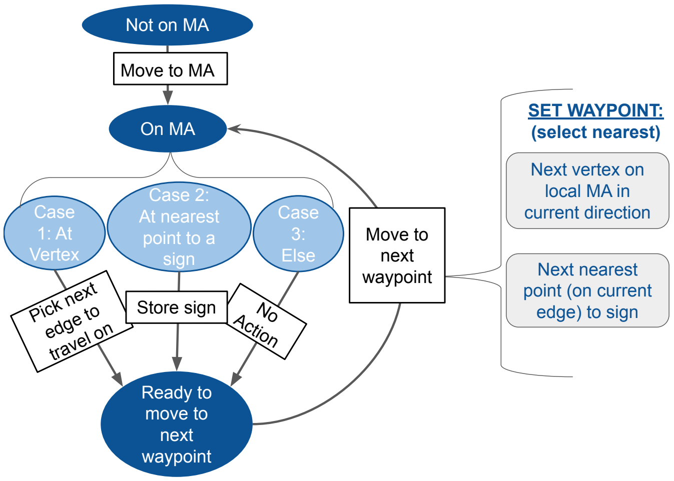
Not on MA
Move to MA
On MA
Case 1: At Vertex
Case 2: At nearest point to a sign
Case 3: Else
Pick next edge to travel on
Store sign
No Action
Ready to move to next waypoint
Move to next waypoint
SET WAYPOINT: (select nearest)
Next vertex on local MA in current direction
Next nearest point (on current edge) to sign

Figure 4. Diagram of robot's navigation logic. Robot uses signs to assign directions to graph edges and makes changes to navigation logic a vertices on the graph and at points where signs can be seen.

- Text and its bounding box
- Arrows and their bounding boxes: each associated to corresponding text as well as the arrow's angle.

The whole system consists of 3 major components (illustrated in Fig. 3) which we discuss in the following:

1) Text Spotting: We implement text spotting with an off-the-shelf pretrained text spotter - CharNet [2] [25]. This component outputs a set of texts and their bounding boxes.

2) Arrow Detection: To perform arrow detection, we first fine-tune a pretrained Mask-RCNN [3] [11] with a FPN-ResNet50 backbone [14] on an airport dataset to localize arrows in an image. Then, for each localized rectangular patch that contains an arrow, we estimate the arrow's angle by computing the angle between the horizontal axis of the patch and the vector connecting the center of the patch and the arrowhead (the arrowhead is estimated by looking at the polygon estimated on the contour of the path). The final output from *arrow detection* is a set of bounding boxes containing an arrow along with the arrow's angle.

3) Text Arrow Grouping: For this version of our perception system we assume each body of text corresponds to exactly one arrow. We group text to its respective arrow symbol by estimating the centroid of each bounding box (both text and arrow), and assign an arrow box to each text box based on nearest centroid. The confidence score for each text-arrow grouping is the product of the confidence score of the arrow detection and the confidence score of the text spotted.

C. Navigation Using Signs

Suppose the robot ρ stores: a list of seen signs along with their sign content and direction assignments for edges on the map skeletonization.

[1] Text spotting is simultaneous localization and recognition of text in a scene

[2] The pretrained model is here: https://github.com/MalongTech/research-charnet. We do not fine-tune the model.

[3] We use the pretrained model from Detectron2 [24] which is here: https://github.com/facebookresearch/detectron2/blob/master/MODEL_ZOO.md

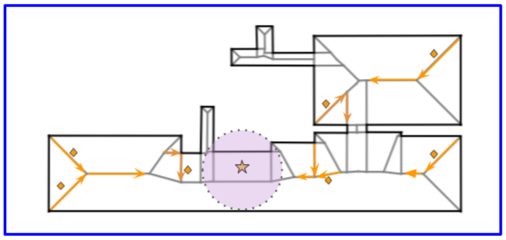

Figure 5. Komatsu Airport (KMQ) with directions, depicted by arrows assigned to their $\Gamma(P)$. The goal, a food court, is depicted with a yellow star, the orange diamonds are signs and the arrows are the directions derived from the signs' interpreted σ. The translucent purple circle around the goal represents the locations where the robot can sense the goal and doubles as a visualization for the size of the robot's sensing sphere.

For each timestep, ρ senses and navigates while interpreting new signs along the way as shown in Fig. 2. A high level illustration of this process is in Fig. 5 and explained in more detail in [13]. As a baseline, the robot travels on a skeletonization of the space, randomly picking a new edge whenever it hits a vertex. We augment the random walk strategy to incorporate the robot's stored odometry data and prevent unnecessary retraversal of previously seen paths. With zero signage, the robot does random walk on an undirected graph without edge retraversal. With signs, the robot can then theoretically improve average trajectory length because signs can be converted to edge direction assignments for edges on the graph.

IV. Evaluation

A. Airport Dataset

We collected a dataset of cellphone camera pictures of airport signs and google streetview captures of signs in airports. All images in the dataset are annotated with bounding box coordinates for text, signs, and arrow symbols, orientation of arrow symbols on signs, and sign text. The dataset contains 240 images which we divide into three sets: training, testing for the vision system, and testing for robot navigation. The first set (199 images) is used to fine-tune arrow detection; the second set (22 images) is used to evaluate the vision system; the last set (13 images) is used to evaluate robot navigation on real airport maps. No airports included in the first set are included in the images contained in the other two sets.

B. Vision System Evaluation

For our initial vision system, we evaluate each component described in Fig. 3 separately and then the system as a whole. A good detector should be able to detect all the ground truths (high recall) without producing too many false positives (high precision). Since most vision components produce a confidence score for their prediction and different confidence thresholds would affect their precision and recall, we report Average Precision (AP) or Area under Precision Recall Curve [8] - a standard object detection metric that summarizes the precision and recall of the detection system at all confidence thresholds. Each set of correctness criteria is explained in its respective subsection.

1) Text Spotting: Text spotted by our model is correct if and only if the produced bounding box intersects with a ground truth bounding box with an Intersection-over-Union (IOU) of at least 0.5 and the text is correctly recognized. For the current version of the system, we evaluate only on a fixed set of predetermined "relevant" words (e.g. "departures") rather than all words in the scene. We report the performance of our text spotter in table I. Without fine-tuning on our collected dataset, our text detector achieves an AP of 47.21%. We also report localization AP which quantifies the text spotter's ability to produce bounding box of interest, regardless of whether the text in the box is correctly recognized. The localization AP is identical to AP, indicating that our text spotter is able to simultaneously localize and recognize text of interest in an image.

	AP (%)	Localization AP (%)
Text Spotting	47.21	47.21
Arrow Detection	4.82	43.82

Table I
Average Precision (AP) and Localization AP of Text Spotting and Arrow Detection.

2) Arrow Detection: We consider arrow detection correct if and only if the generated bounding box intersects with a ground truth text box with an IOU of at least 0.5 and the predicted angle of the arrow is within 5 degrees of the ground truth angle [4]. The AP and localization AP are reported in table I. Unlike the text spotter, our arrow detector is able to localize the arrows of interest (with an AP of 43.82%) but fails at estimating the orientation of the arrow (with an AP of 4.82 %)

3) Text-arrow Grouping: We consider a predicted grouping correct if and only if each of the two bounding boxes (text and arrow) of the predicted group overlap with the corresponding bounding boxes of a ground truth with an IOU of at least 50% each. Since an arrow detector with low precision would significantly exacerbate the performance of grouping, we filter out all the arrows with confidence score less than 0.9 (this threshold is found using cross validation). Our simple grouping function is able to achieve an AP of 15.87%

4) Whole System: A predicted sign is correct if and only if

- The predicted text box overlaps the ground truth text box with an IOU of at least 0.5 and the text in the two boxes match
- The predicted arrow box overlaps the ground truth arrow box with an IOU of at least 0.5 and the orientations of the arrows are within 5 degrees of each other.

Our system achieves an AP of 11.58%. Selected output of our vision system can be found in fig 6.

[4] The tolerable amount of error for angle estimation is dependent on each environment for each navigation task. For evaluation, we pick a conservative margin of 5 degrees for simplicity and convenience.

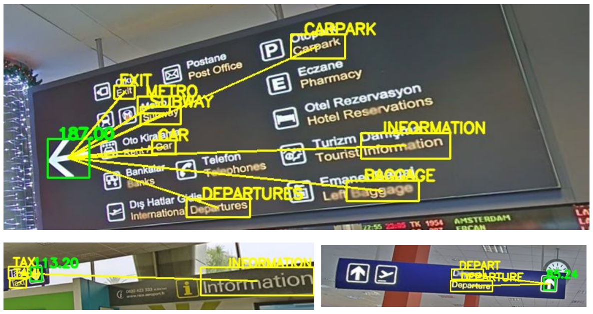

Figure 6. Several examples of output images from our vision system. Bounding boxes for text are in yellow, bounding boxes for arrows are in green. Lines illustrate groupings of text with respective arrows, and the number value is the angle orientation of the detected arrow.

C. Demonstration of Navigation Using Simulated Robot in Real Airport Maps

We demonstrate a simulation of a robot using our vision system in a real airport map. We test in three airports: Lyon–Saint-Exupéry Airport (LYS) (6 signs pointing to the train station), José María Córdova International Airport (MDE) (4 signs pointing to domestic departures), Montpellier–Méditerranée Airport (MPL) (3 signs pointing to baggage claim). For each of these airports we have collected human-labeled sign interpretations which we consider to be our ground truth comparison for the perception system, we only look at signs and arrows relevant to each airport's fixed goal destination. The directions generated by the vision system for LYS result in the same robot navigation behavior with the ground truth sign directions for 80% of the signs. The directions generated by the vision system for MDE align 50% with the ground truth sign directions. The directions generated by the vision system for MPL align 67% with the ground truth sign directions. We use the navigation logic is as detailed in Fig. 4 to compare trajectory lengths of a robot using no signs (an improved random walk as done in [13] and explained in Section III-C) versus the signs identified using our perception system. Each airport has 1000 trials, with a fixed goal point and signs but random starting locations. The results are included in Table II.

Airport Code	Average Trajectory Length Without Signs	Average Trajectory Length With Signs	% Improvement
LYS	1041.72	802.12	23%
MDE	2691.83	2557.24	5%
MPL	800.10	664.083	17%

Table II
COMPARISON OF TRAJECTORY LENGTHS USING NO SIGNAGE AND USING SIGNAGE INTERPRETED BY OUR PERCEPTION SYSTEM.

V. DISCUSSION

A. Failure Cases of Perception System

Our system has difficulty detecting small objects of interest. In the top and bottom left image in fig 7, we observe that none of the arrows are detected; some of the small text such as taxis and arrivals are also not detected in the bottom right image in fig 7. Another common failure we observe is the error in arrow angle estimation (See the bottom right image in figure 7 and bottom left image in figure 6). The template matching solution we have currently is not robust against small variations in arrow shape.

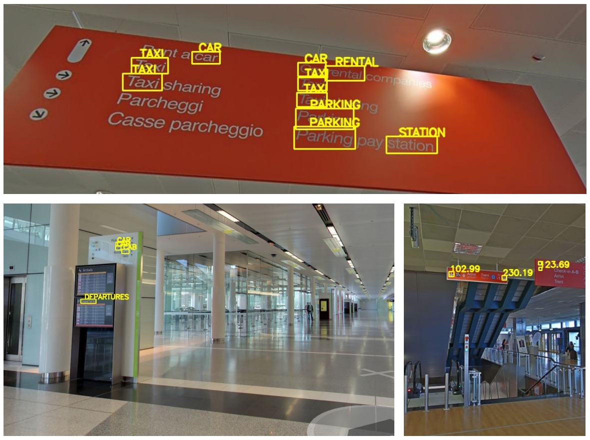

Figure 7. A selection of failure case outputs from our perception system. Failures often occur when text and arrows are very small relative to sign size. Text and bounding boxes are in yellow.

B. Next Steps

1) Arrow Angle Estimation: To improve arrow angle estimation, we propose to extend the box regressor of Mask-RCNN to also predict the arrow angle.

2) Demonstration of Navigation: Each improperly interpreted sign can have a large impact on the improvement signs provide to overall trajectory length. For example, for MDE, we get an improvement of only 5%, however there were only four signs in the space, and only half of them were interpreted with valid directions. Both of these signs were positioned quite close to the goal point for the robot, and thus the signs were unable to provide great improvement in navigation performance.

We would expect that high recall from our perception system would improve trajectory efficiency in navigation, and the results from the three airports we use in our simulated navigation demonstration align with this trend. These findings suggest that we should extend simulation to more airports to investigate our hypothesis further. We also intend to incorporate more airport images that we have collected into our training set.

Beyond improving the performance of our perception system on curated images, we want to show how our system works on a real robot deployed in the real world. Next steps for our demonstration on a real robot include using the robot's limited on-board sensing to do sign detection and interpretation for real-time navigation on the fly.

ACKNOWLEDGMENTS

This work was partly supported by a DARPA LwLL grant. We are grateful for this support.

References

[1] In *Proceedings of the 2017 acm/ieee international conference on human-robot interaction*, pages 283–292, 2017.

[2] Vavilin Andrey and Kang Hyun Jo. Automatic detection and recognition of traffic signs using geometric structure analysis. In *2006 SICE-ICASE International Joint Conference*, pages 1451–1456. IEEE, 2006.

[3] Michal Busta, Lukas Neumann, and Jiri Matas. Deep textspotter: An end-to-end trainable scene text localization and recognition framework. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2204–2212, 2017.

[4] Jingwei Cao, Chuanxue Song, Silun Peng, Feng Xiao, and Shixin Song. Improved traffic sign detection and recognition algorithm for intelligent vehicles. *Sensors*, 19(18):4021, 2019.

[5] Konstantinos Charalampous, Ioannis Kostavelis, and Antonios Gasteratos. Robot navigation in large-scale social maps: An action recognition approach. *Expert Systems with Applications*, 66:261–273, 2016.

[6] Hyeong In Choi, Sung Woo Choi, and Hwan Pyo Moon. Mathematical theory of medial axis transform. *pacific journal of mathematics*, 181(1):57–88, 1997.

[7] Howie Choset and Keiji Nagatani. Topological simultaneous localization and mapping (slam): toward exact localization without explicit localization. *IEEE Transactions on Robotics and Automation*, 17(2):125–137, 2001.

[8] Mark Everingham, Luc Van Gool, Christopher KI Williams, John Winn, and Andrew Zisserman. The pascal visual object classes (voc) challenge. *International journal of computer vision*, 88(2):303–338, 2010.

[9] Ross Girshick. Fast r-cnn. In *Proceedings of the IEEE international conference on computer vision*, pages 1440–1448, 2015.

[10] Ross Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 580–587, 2014.

[11] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask r-cnn. In *Proceedings of the IEEE international conference on computer vision*, pages 2961–2969, 2017.

[12] Hui Li, Peng Wang, and Chunhua Shen. Towards end-to-end text spotting with convolutional recurrent neural networks. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 5238–5246, 2017.

[13] Claire Liang, Florian T Pokorny, and Ross A Knepper. No map, no problem: A local sensing approach for navigation in man-made spaces using signs. In *Under Review for 2020 International Conference on Intelligent Robots and Systems*, 2020.

[14] Tsung-Yi Lin, Piotr Dollár, Ross Girshick, Kaiming He, Bharath Hariharan, and Serge Belongie. Feature pyramid networks for object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2117–2125, 2017.

[15] Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, Cheng-Yang Fu, and Alexander C Berg. Ssd: Single shot multibox detector. In *European conference on computer vision*, pages 21–37. Springer, 2016.

[16] Xuebo Liu, Ding Liang, Shi Yan, Dagui Chen, Yu Qiao, and Junjie Yan. Fots: Fast oriented text spotting with a unified network. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5676–5685, 2018.

[17] Pengyuan Lyu, Minghui Liao, Cong Yao, Wenhao Wu, and Xiang Bai. Mask textspotter: An end-to-end trainable neural network for spotting text with arbitrary shapes. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 67–83, 2018.

[18] Joseph Redmon and Ali Farhadi. Yolo9000: better, faster, stronger. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 7263–7271, 2017.

[19] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. You only look once: Unified, real-time object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 779–788, 2016.

[20] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In *Advances in neural information processing systems*, pages 91–99, 2015.

[21] Safat B Wali, Mahammad A Hannan, Aini Hussain, and Salina A Samad. An automatic traffic sign detection and recognition system based on colour segmentation, shape matching, and svm. *Mathematical Problems in Engineering*, 2015, 2015.

[22] Matthew R Walter, Sachithra Hemachandra, Bianca Homberg, Stefanie Tellex, and Seth Teller. Learning semantic maps from natural language descriptions. In *Robotics: Science and Systems*, 2013.

[23] Yuan Wei, Emma Brunskill, Thomas Kollar, and Nicholas Roy. Where to go: Interpreting natural directions using global inference. In *2009 IEEE International Conference on Robotics and Automation*, pages 3761–3767. IEEE, 2009.

[24] Yuxin Wu, Alexander Kirillov, Francisco Massa, Wan-Yen Lo, and Ross Girshick. Detectron2. https://github.com/facebookresearch/detectron2, 2019.

[25] Linjie Xing, Zhi Tian, Weilin Huang, and Matthew R Scott. Convolutional character networks. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 9126–9136, 2019.