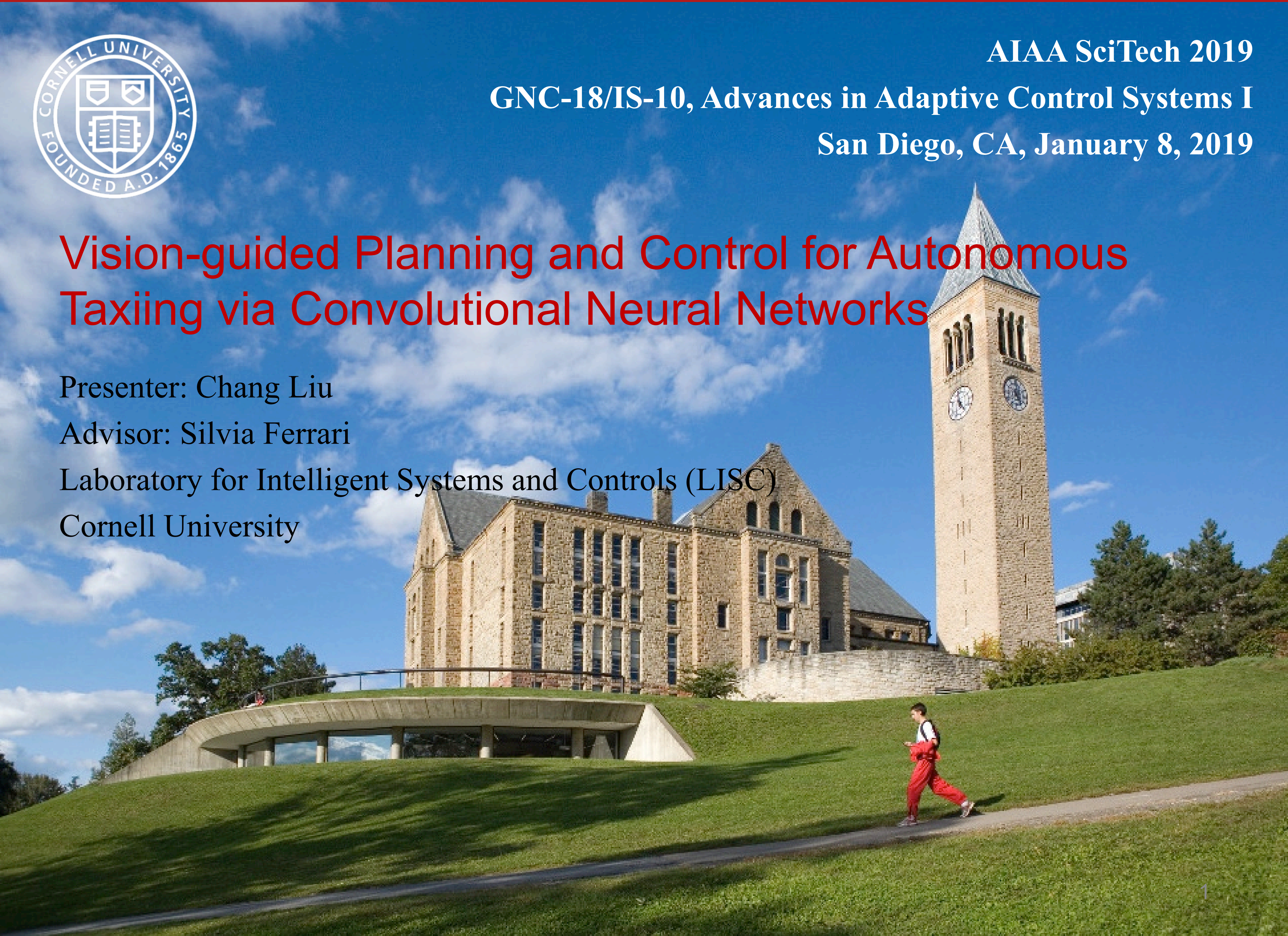


AIAA SciTech 2019

GNC-18/IS-10, Advances in Adaptive Control Systems I

San Diego, CA, January 8, 2019

Vision-guided Planning and Control for Autonomous Taxiing via Convolutional Neural Networks

Presenter: Chang Liu

Advisor: Silvia Ferrari

Laboratory for Intelligent Systems and Controls (LISC)

Cornell University

Outline

- Motivation and Background
- Problem Formulation
- Technical Approach:
  - Object Recognition using Mask-RCNN
  - ATC-based Path Planning
  - Hybrid Control of Autonomous Taxiing Aircraft
- Simulation Results
- Conclusions

Motivation and Background

Motivation

- Aerodromes are becoming increasingly complex and crowded
- Dangerous situations seriously affects aerodrome safety

Crowded airports

Runway incursion incidents

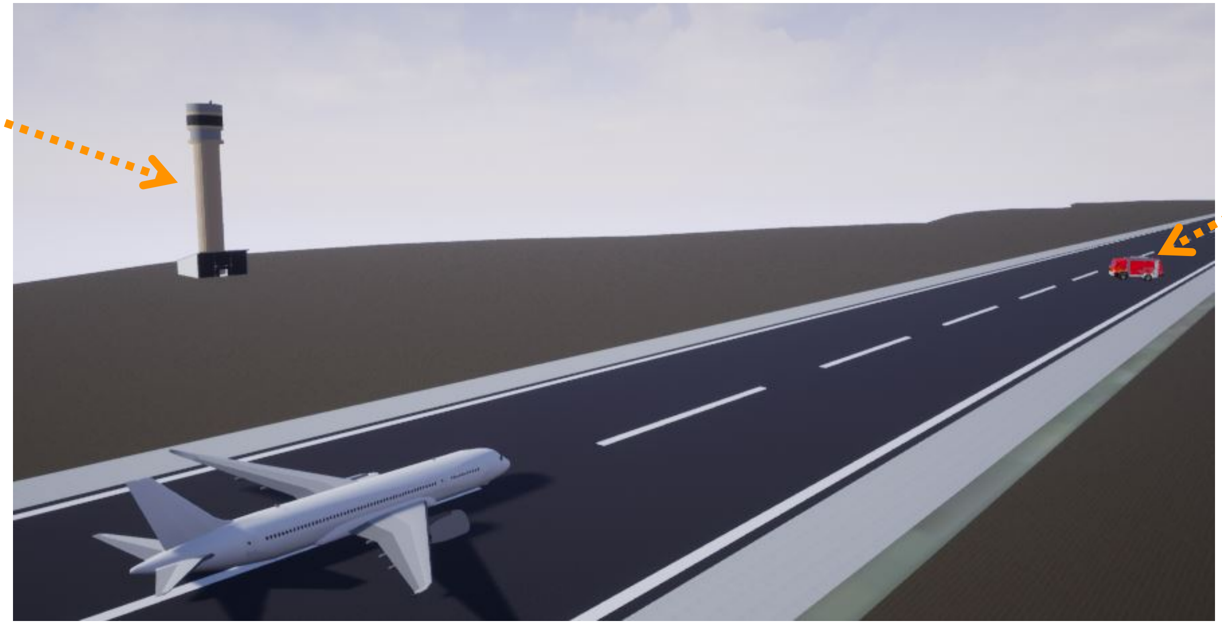
Incidents of runway incursions in 2018 (from FAA)

Year 2018	Operational Incident	Pilot Deviation	Vehicle Pedestrian Deviation	Other	Total
Totals	345	1142	335	10	1832

Motivation

Autonomous Taxiing:

- Automating aircraft taxiing process without human intervention
  - Take Air Traffic Control (ATC) commands in the loop
  - Detect obstacles and unforeseen conditions *in situ*
  - Generate corrective planning and control to guarantee aircraft safety



ATC tower

Unforeseen objects

Related Work

Surface Operation Scheduling: schedule the use of taxiways, runways, and gates to reduce the overall travel time and maximize throughput (Morris 2016)

Aircraft Path Planning and Control: Generate energy-efficient and collision-free trajectories based on aircraft motion models (McGee 2007, Coetzee 2011, Chen 2016, Zhang 2018)

Situation Awareness for Planning: Uses machine learning algorithms for taxiway feature extraction and unknown obstacle identification (Lu 2016, Lu 2018)

Limitations:
- Lack of ability to incorporate ATC commands
- Planning and control ignores environmental perception
- Unable to handle unexpected dangerous situations

Problem Formulation

Problem Formulation

- **Goal**: Develop a vision-guided path planning and control approach for autonomous taxiing under both normal conditions and unforeseen conditions

- Hybrid system modeling:

$$\mathbf{s}(k+1) = \mathbf{f}_\xi[\mathbf{s}(k), \mathbf{u}_\xi(k)] \quad \xi(k+1) = \mu(k)$$

- Objective function

$$J = \varphi(\mathbf{s}(k_f)) + \sum_{j=k}^{k_f-1} \Psi_\xi(\mathbf{s}(j), \mathbf{u}_\xi(j), \mu(j))$$

- Continuous controller $\mathbf{u}_\xi(k)$

- Discrete controller $\mu(k) \in \mathbb{Z}_+$

Normal conditions

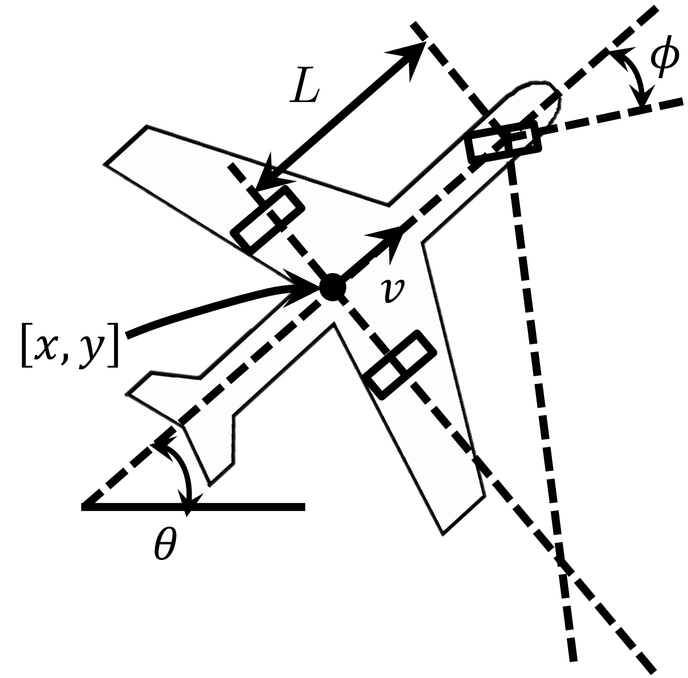
Unforeseen conditions

$\xi \in \mathbb{Z}_+$	System discrete mode	$\mathbf{s}$	System continuous state
J	Objective function	$\mathbf{f}$	Continuous dynamics
$[k, k_f]$	Planning horizon	φ, Ψ	Stage cost

Aircraft Motion Model

- The aircraft uses a *simple car* model, defined as,

$$\begin{bmatrix} \mathbf{q}_e(k+1) \\ \theta(k+1) \\ \upsilon(k+1) \end{bmatrix} = \begin{bmatrix} \mathbf{q}_e(k) \\ \theta(k) \\ \upsilon(k) \end{bmatrix} + \begin{bmatrix} \upsilon(k)\cos\theta(k) \\ \upsilon(k)\sin\theta(k) \\ \frac{\upsilon(k)}{L}\tan\phi(k) \\ \beta(k) \end{bmatrix} \Delta T$$



The simple car model of an aircraft

- Aircraft state $\mathbf{s}(k) = [\mathbf{q}_e(k) \quad \theta(k) \quad \upsilon(k)]^T$
- Coordinate of the rear-axle center $\mathbf{q}_e = [x, y]^T$
- Aircraft control input $\mathbf{u}(k) = [\phi(k), \beta(k)]^T$

θ	Yaw angle	υ	Speed
ϕ	Steering angle	β	Acceleration
ΔT	Sampling interval	L	Front-rear axle distance

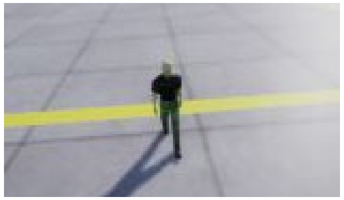
Camera Measurement Model

- Camera measurement model $\mathbf{z}(k) = [l(k), d_l(k)]^T$
  - objects classification $l(k) \in \{0,1,2,\ldots\}$
  - camera-to-object distance $d_l(k) \geq 0$
- Recognizing three semantic classes of incursion objects
  - "People" $l(k) = 1$
  - "Animals" $l(k) = 2$
  - "Ground Vehicles" $l(k) = 3$

  $l(k) = 0$ means no detected object

Onboard RGB-D Camera

Camera View



People

Animal

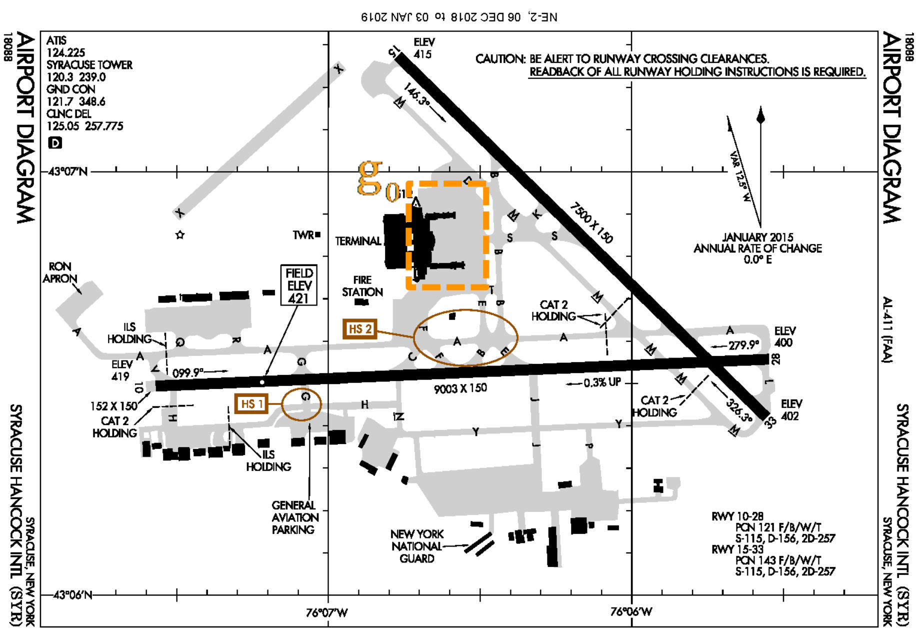
Vehicle

Airport Modeling

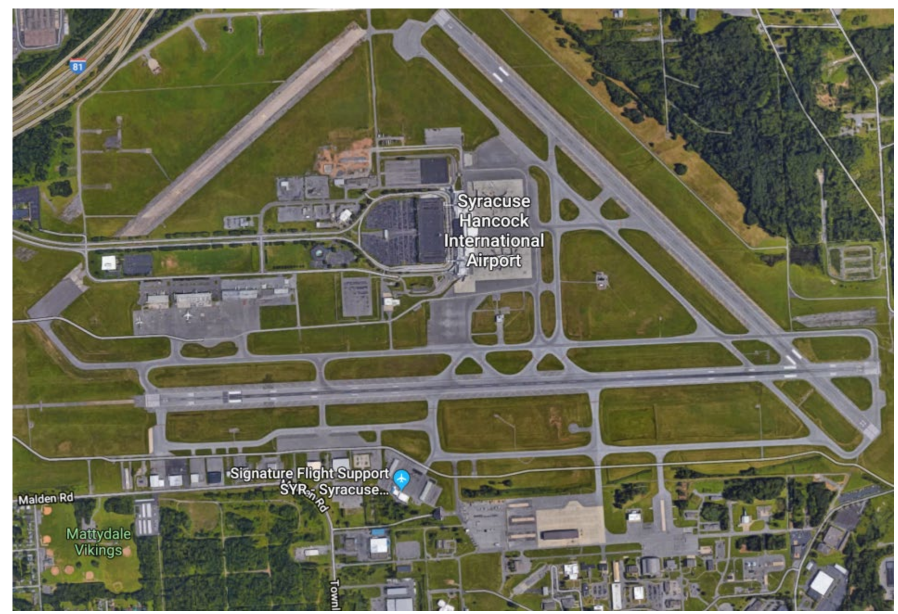
- Taxiway label set $L_{\mathbf{A}} = \{A, E, P, F, V, \ldots\}$
- Runway label set $L_{\mathbf{P}} = \{10, 28, 15, 33\}$
- Terminal set $L_{\mathbf{G}} = \{g_0, g_1, g_3, g_{10}, g_{22}, \ldots\}$

  $g_i, i = 1,2,\ldots$ represents the ith gate

- Taxiway region $W_a \subseteq R^2,\ a \in L_{\mathbf{A}}$
- Runway region $W_p \subseteq R^2,\ p \in L_{\mathbf{P}}$
- Terminal region $W_{g_0} \subseteq R^2$



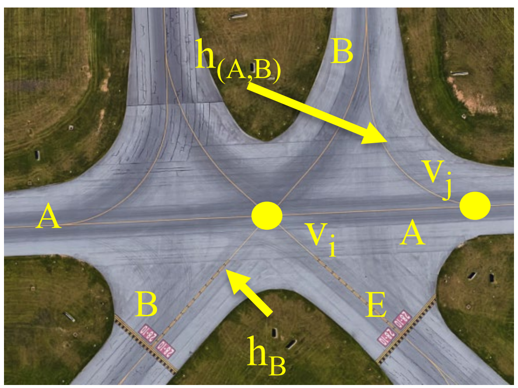
Airport Diagram

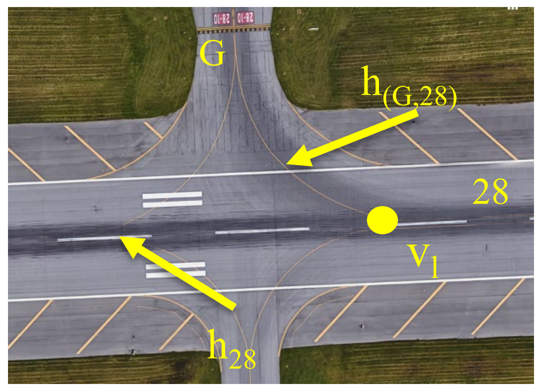


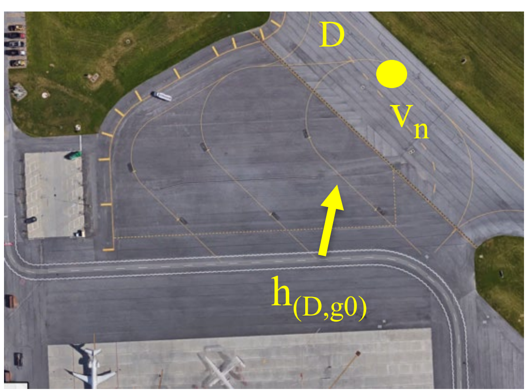
Airport Satellite Map

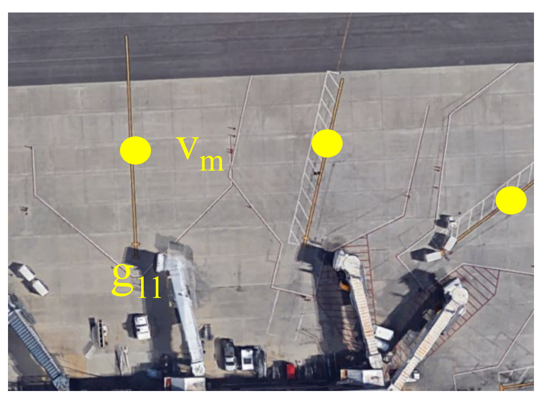
Airport Modeling

- Taxiway centerline $h_a(\mathbf{q}) = 0,\ \ a \in L_{\mathbf{A}}$
- Runway centerline $h_p(\mathbf{q}) = 0,\ \ p \in L_{\mathbf{P}}$
- Taxiway connecting arc $h_A(\mathbf{q}) = 0,\ a \in L_{\mathbf{A}} \times L_{\mathbf{A}}$
- Taxiway-runway arc $h_P(\mathbf{q}) = 0,\ P \in L_{\mathbf{A}} \times L_{\mathbf{P}}$
- Taxiway-terminal arc $h_T(\mathbf{q}) = 0,\ T \in L_{\mathbf{A}} \times \{g_0\}$









Nodes $v = (\gamma, \mathbf{q})$,
where $\gamma \in L_{\mathbf{A}} \times L_{\mathbf{P}} \times L_{\mathbf{G}}$

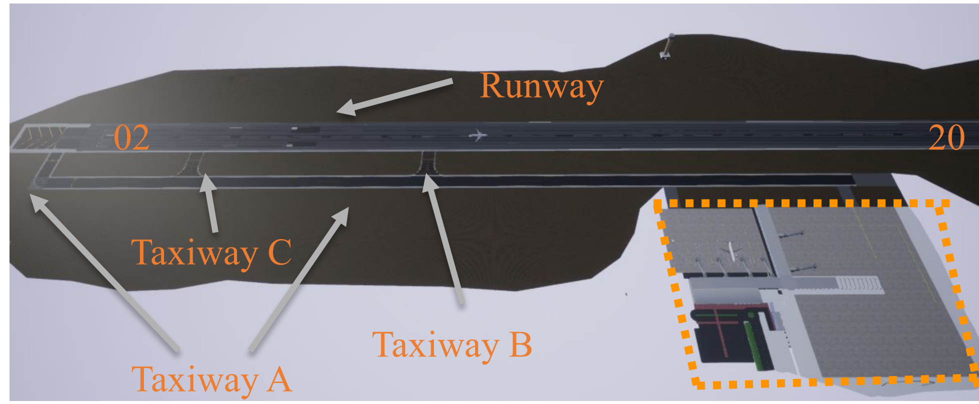
- connection of two regions
- terminal gates
- aircraft current position

Example nodes:

$$v_i = \{A, B, E\}$$
$$v_j = \{A, B\}$$
$$v_l = \{G, 28\}$$
$$v_m = \{g_{11}, g_0\}$$
$$v_n = \{D, g_0\}$$

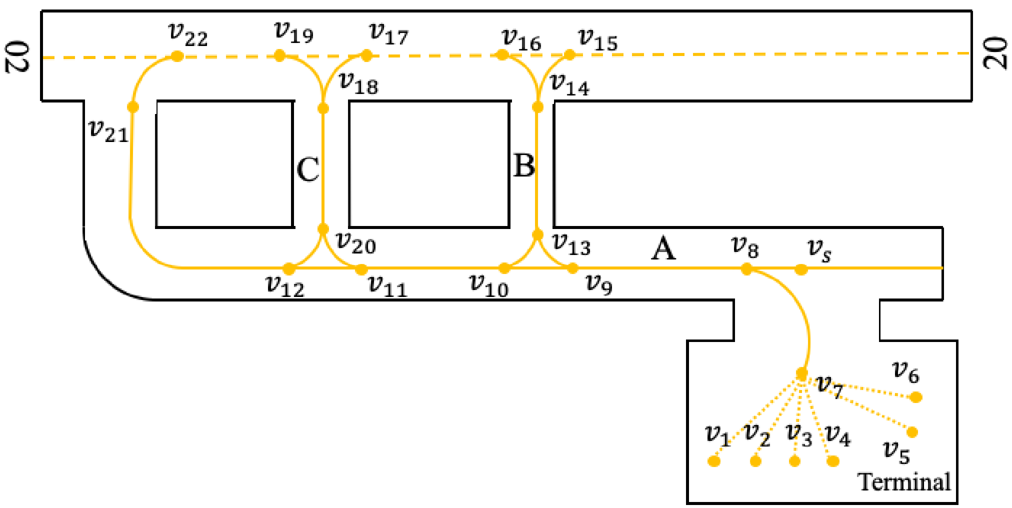
Airport Modeling

- **Airport Graph:** a topological graph $G = (V, \Xi)$ representing the connectivity of different airport regions.
  - Node set $V = \{v\}$
  - Edge set $\Xi = \{v_i \cap v_j \mid v_i, v_j \in V\}$

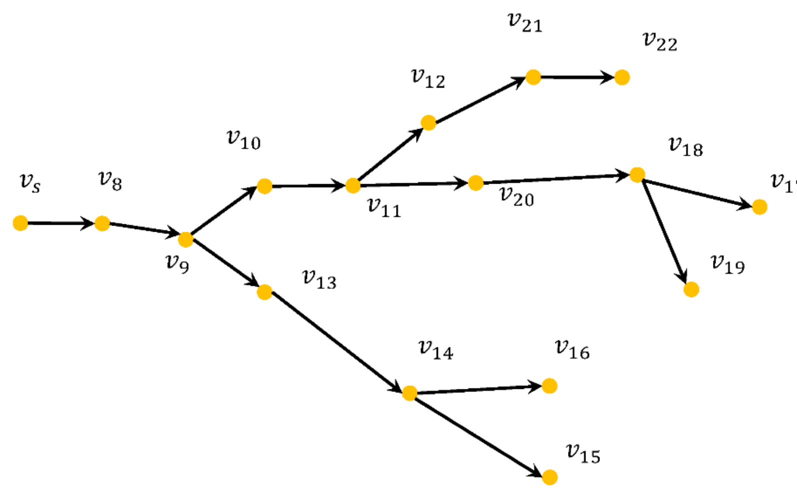


Terminal gate region

Simulated airport



Nodes, centerlines, and connecting arcs



Partial graph of the airport

ATC Commands

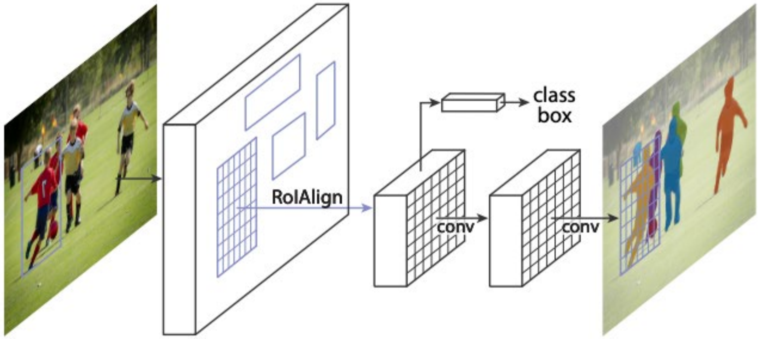
Command Category c	Instruction	ATC Command Examples
Cruising (c_1)	Move along certain taxiways to a runway.	• "Runway Three-Six Left, taxi via Taxiway Alpha, hold short of Taxiway Charlie." • "Cross Runway One-Six Left and Runway One-Six Right at Taxiway Bravo."
Traffic Following (c_2)	Follow the traffic.	• "Follow (traffic), cross Runway Two-Seven Right, at Taxiway Whiskey."
Holding (c_3)	Hold short of a runway or hold in position.	• "Hold short of runway Two-Seven." • "Hold in position."

Technical Approach

Object Recognition using Mask-RCNN

Mask-RCNN: state-of-the-art object detector

- Consists of a region proposal network and a binary mask classifier
- Uses RGB images as input
- Outputs a class label, bounding box, segmentation mask, and a confidence level



The Mask-RCNN structure (He 17')

Recognition results using Mask-RCNN

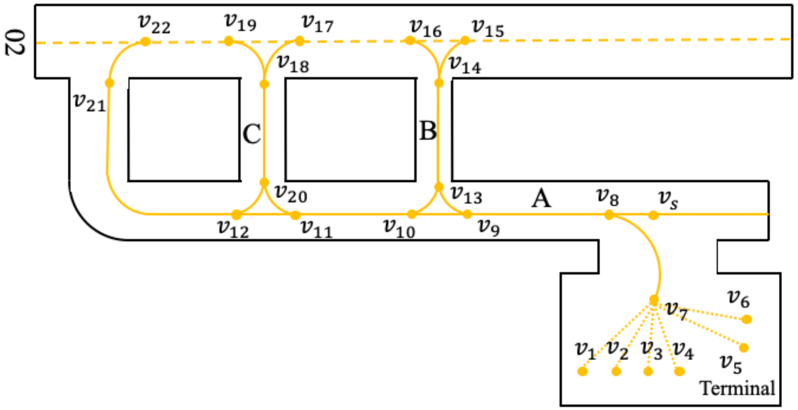
ATC-based Path Planning

- ATC command $\Psi = (\psi_1, \psi_2, \ldots, \psi_{n-1}, \psi_n)$ where

  $\psi_1, \psi_2, \ldots, \psi_{n-1} \in L_{\mathbf{A}}$ and $\psi_n \in L_{\mathbf{A}} \cup L_{\mathbf{P}} \cup L_{\mathbf{G}}$

  - Example: "Runway Two-Zero, taxi via Taxiway Alpha and Bravo"

    $\Rightarrow \Psi = (A, B, 20)$

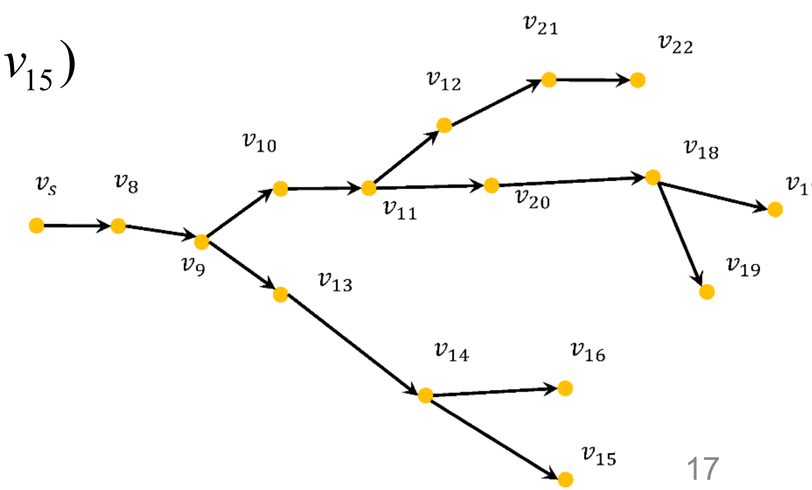


- Path Generation from ATC Commands
  - Connect graph nodes in accordance with ATC commands

    Example: node sequence $(v_s, v_8, v_9, v_{13}, v_{14}, v_{15})$

  - Turn node sequence into waypoints

    $$\mathbf{q}^* = [(x_r(1), y_r(1))^T, \ldots, (x_r(N), y_r(N))^T]$$



(x_r, y_r)	reference waypoint
Ψ	ATC command sequence

Hybrid Control of Autonomous Aircraft

- Five aircraft modes

$$\xi = \{\text{cruising (1), traffic following (2), holding (3), incursion (4), idle (5)}\}$$

- The optimization problem in mode ξ is defined as

$$\begin{aligned}
\min_{\mathbf{x}_\xi} \quad & J_\xi = \varphi\big(x(k_f), y(k_f), \upsilon(k_f)\big) + \sum_{j=k}^{k_f-1} \Psi\big(x(j), y(j), \upsilon(j), \mathbf{u}_\xi(j)\big) \\
s.t. \quad & \mathbf{s}(k) = \mathbf{s}_0 \\
& \mathbf{s}(j+1) = \mathbf{f}(\mathbf{s}(j), \mathbf{u}_\xi(j)), \quad j = k, \ldots, k_f - 1 \\
& \mathbf{s}(j) \in S, \quad j = k, \ldots k_f - 1 \\
& \mathbf{u}_\xi(j) \in U_\xi, \quad j = k, \ldots k_f - 1
\end{aligned}$$

where: $\mathbf{x}_\xi = [\mathbf{s}^T(k), \ldots, \mathbf{s}^T(k_f), \mathbf{u}_\xi(k), \ldots \mathbf{u}_\xi(k-1)]^T$ is the optimization variable

(x_r, y_r)	Reference waypoint	$\mathbf{s}_0$	Aircraft initial state
S	State feasible set	U_ξ	Control feasible set

Continuous State Control

- Cruising mode: Follow the reference path and maintain a desired cruise speed υ_c

$$J_1 = \left\| [x(k_f), y(k_f), \upsilon(k_f)]^T - [x_r(k_f), y_r(k_f), \upsilon_c]^T \right\|_2^2$$
$$+ \sum_{j=k}^{k_f-1} \left\| [x(j), y(j), \upsilon(j)]^T - [x_r(j), y_r(j), \upsilon_c]^T \right\|_2^2 + \left\| \mathbf{u}_1(j) \right\|_2^2$$

- Traffic following mode: Follow the reference path and maintain a speed similar to the aircraft in front υ_f

$$J_2 = \left\| [x(k_f), y(k_f), \upsilon(k_f)]^T - [x_r(k_f), y_r(k_f), \upsilon_f]^T \right\|_2^2$$
$$+ \sum_{j=k}^{k_f-1} \left\| [x(j), y(j), \upsilon(j)]^T - [x_r(j), y_r(j), \upsilon_f]^T \right\|_2^2 + \left\| \mathbf{u}_2(j) \right\|_2^2$$

- Holding mode: Decelerate to stop at the hold-short position

$$J_3 = \left\| [x(k_f), y(k_f), \upsilon(k_f)]^T - [x_r(k_f), y_r(k_f), 0]^T \right\|_2^2$$
$$+ \sum_{j=k}^{k_f-1} \left\| [x(j), y(j), \upsilon(j)]^T - [x_r(j), y_r(j), 0]^T \right\|_2^2 + \left\| \mathbf{u}_3(j) \right\|_2^2$$

with the additional constraint $\upsilon(k_f) = 0$

Continuous State Control

- Incursion mode: Decelerate to avoid collision with the incursion object

$$J_4 = \left\| [x(k_f), y(k_f), \upsilon(k_f)]^T - [x_r(k_f), y_r(k_f), 0]^T \right\|_2^2$$
$$+ \sum_{j=k}^{k_f - 1} \left\| [x(j), y(j), \upsilon(j)]^T - [x_r(j), y_r(j), 0]^T \right\|_2^2 + \left\| \mathbf{u}_4(j) \right\|_2^2$$

with the additional *collision avoidance* constraint

$$\left\| [x(j), y(j)]^T - [x_o, y_o]^T \right\|_2^2 \geq d_s, \quad j = k, \ldots k_f - 1$$

- Idle mode: Remain current state

$$\mathbf{u}_5 = \mathbf{0}$$

(x_o, y_o)	Object position
d_s	Safety distance

Discrete State Control Definition

- Cruising mode

$$\mu(k) = \begin{cases} \mu_2 & \text{if } l(k) = 0 \text{ and } c(k) = c_2 \\ \mu_3 & \text{if } l(k) = 0 \text{ and } c(k) = c_3 \\ \mu_4 & \text{if } l(k) \in \{1,2,3\} \\ \mu_1 & \text{otherwise} \end{cases}$$

- Traffic following mode

$$\mu(k) = \begin{cases} \mu_1 & \text{if } l(k) = 0 \text{ and } c(k) = c_1 \\ \mu_3 & \text{if } l(k) = 0 \text{ and } c(k) = c_3 \\ \mu_4 & \text{if } l(k) \in \{1,2,3\} \\ \mu_5 & \text{if } l(k) = 0 \text{ and } \upsilon(k) = 0 \\ \mu_2 & \text{otherwise} \end{cases}$$

- Idle mode

$$\mu(k) = \begin{cases} \mu_1 & \text{if } c(k) = c_1 \\ \mu_2 & \text{if } c(k) = c_2 \\ \mu_5 & \text{otherwise} \end{cases}$$

- Holding mode

$$\mu(k) = \begin{cases} \mu_1 & \text{if } l(k) = 0 \text{ and } c(k) = c_1 \\ \mu_2 & \text{if } l(k) = 0 \text{ and } c(k) = c_2 \\ \mu_4 & \text{if } l(k) \in \{1,2,3\} \\ \mu_5 & \text{if } l(k) = 0 \text{ and } \upsilon(k) = 0 \\ \mu_3 & \text{otherwise} \end{cases}$$

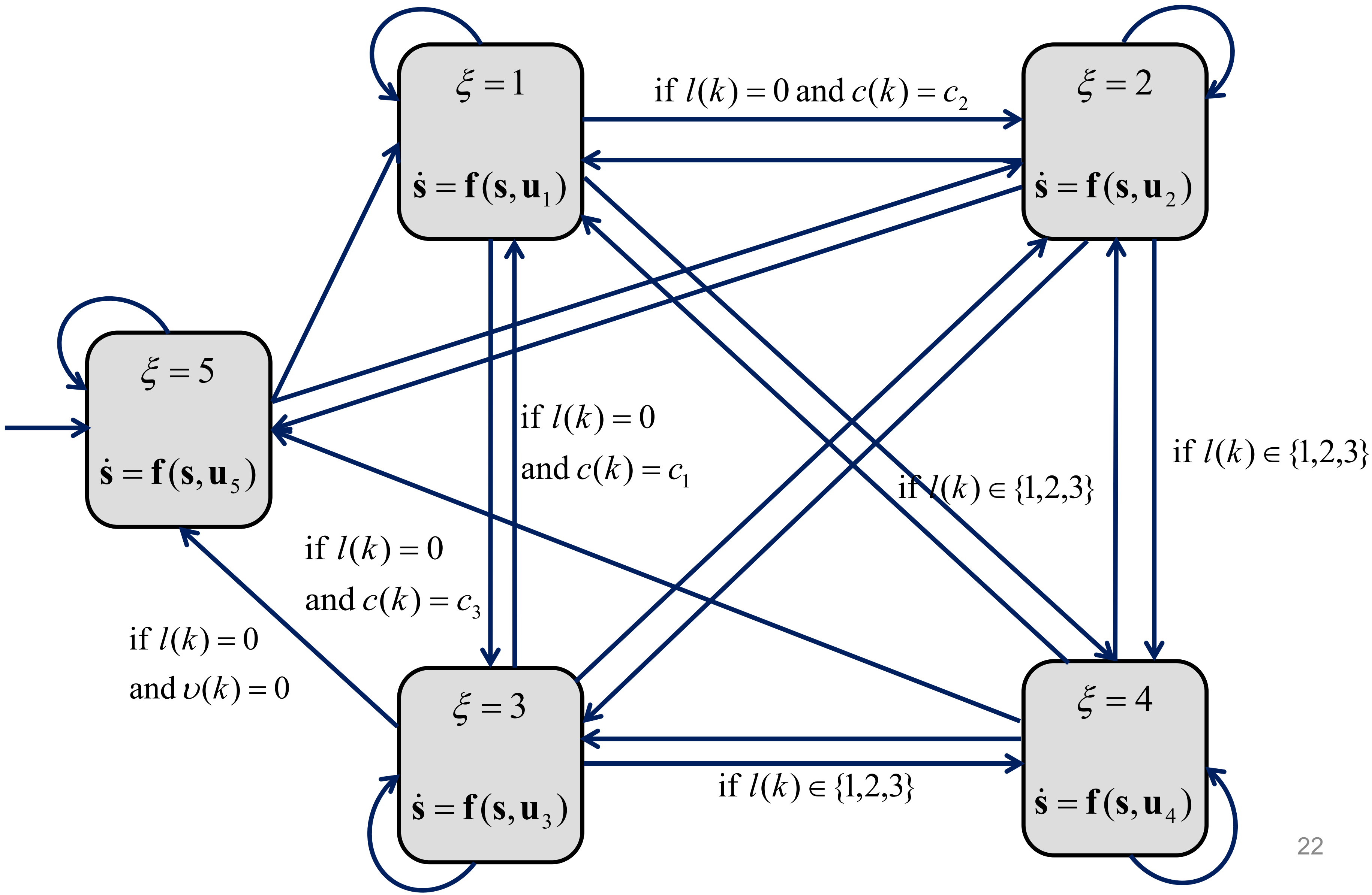
- Incursion mode

$$\mu(k) = \begin{cases} \mu_1 & \text{if } c(k) = c_1 \\ \mu_2 & \text{if } c(k) = c_2 \\ \mu_3 & \text{if } c(k) = c_3 \\ \mu_5 & \text{if } \upsilon(k) = 0 \\ \mu_4 & \text{otherwise} \end{cases}$$

where: $\mu(k) \in \mathbb{Z}_+$ and $\mu_i = i$

c_1, c_2, c_3	ATC command category
$l(k)$	Object classification

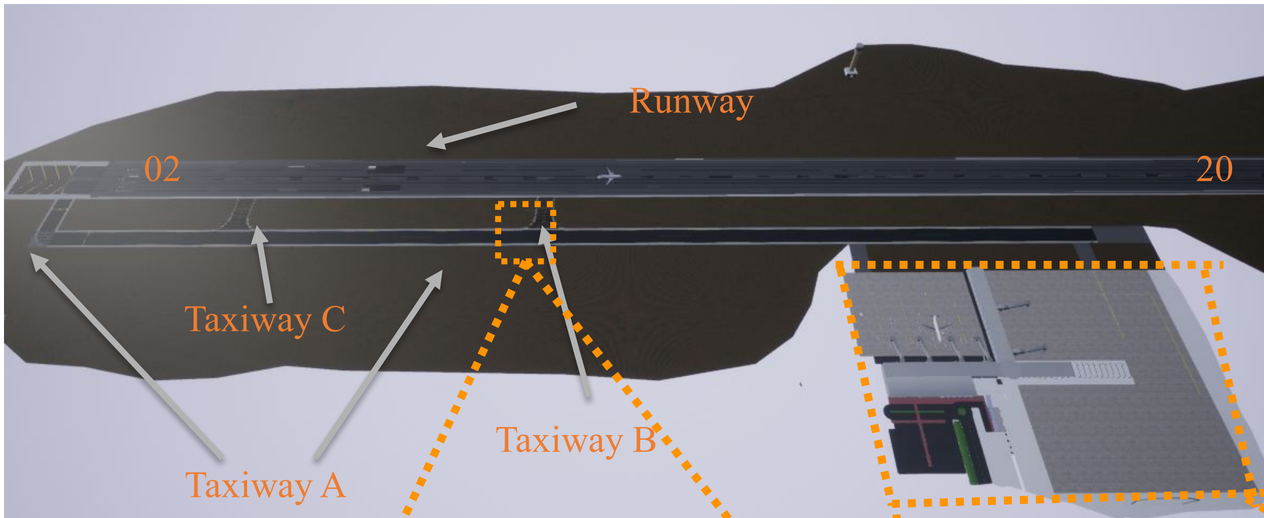
Visualization of Discrete State Control



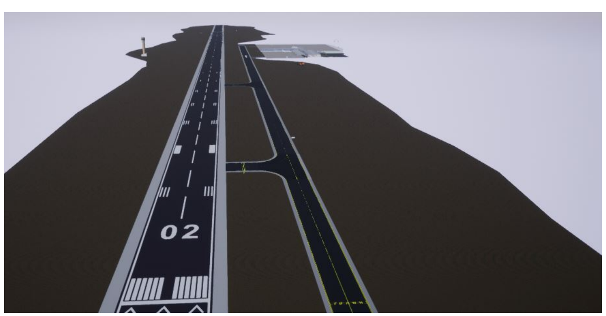
Simulation Results

Simulation Setup

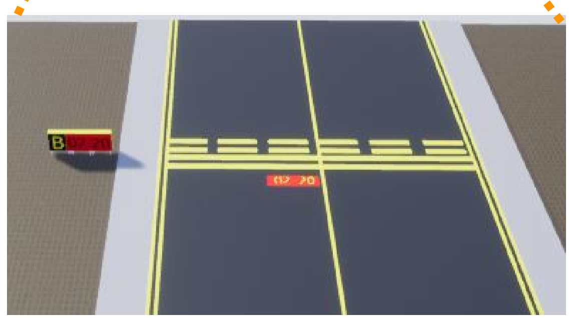
- A simulated small-sized airport modeled in UnrealEngine®



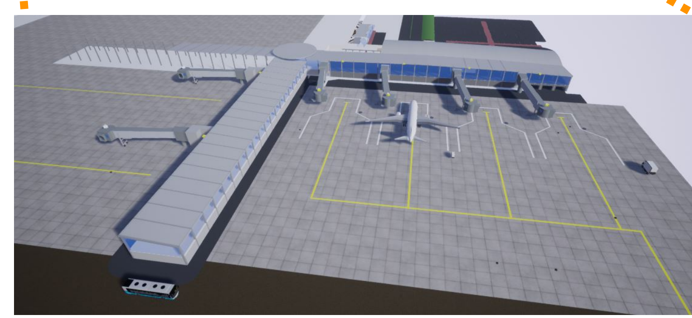
Simulated airport



Runway and taxiways



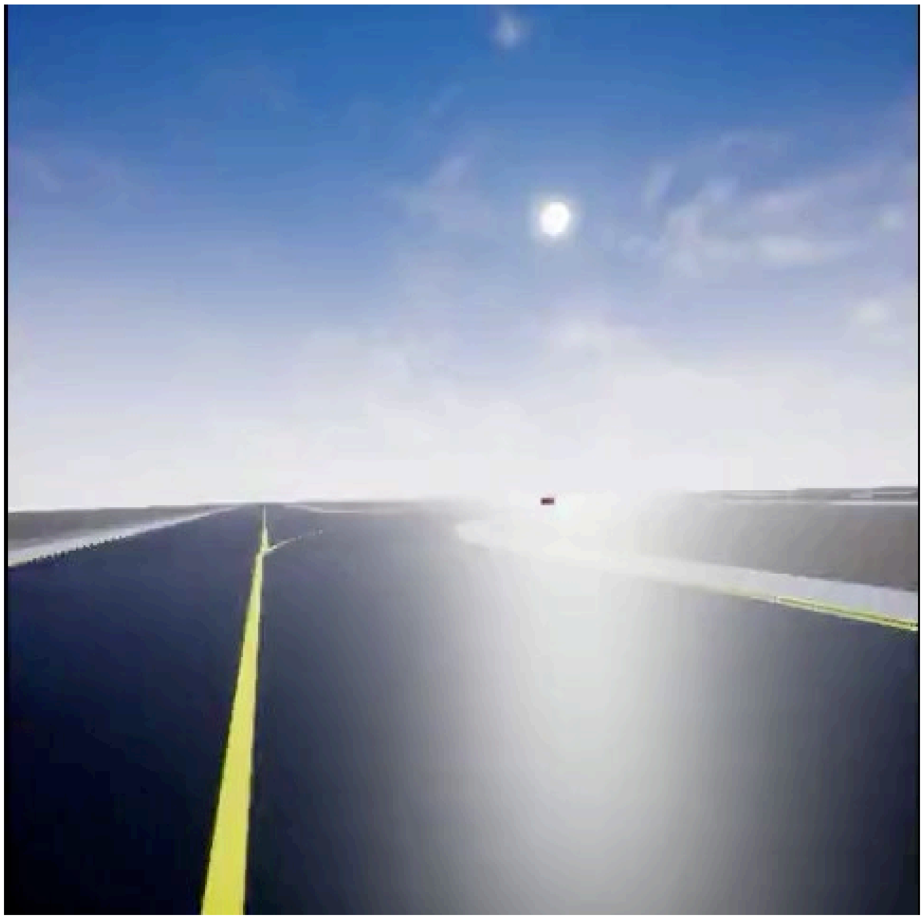
Ground markings and signs

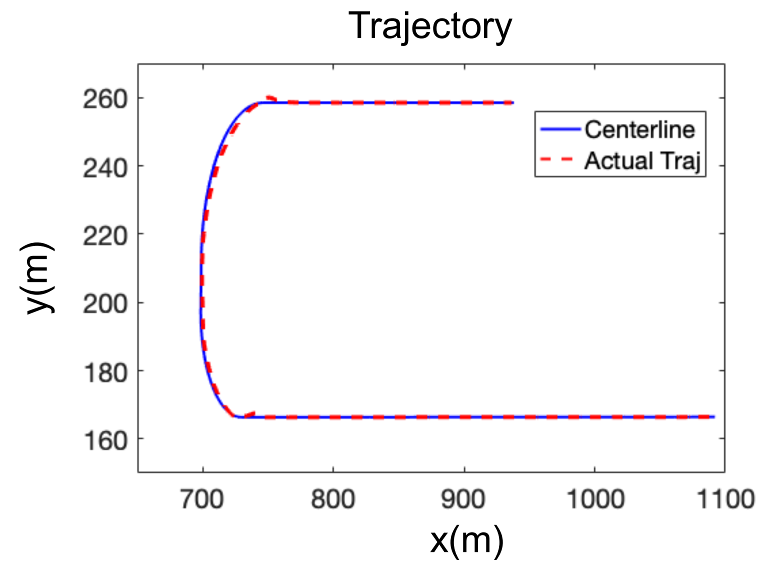


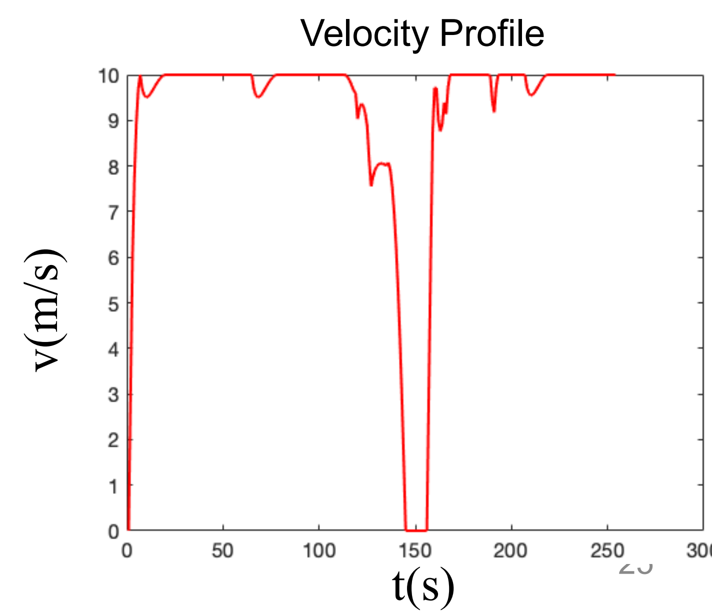
Terminal area

Results – Normal Taxiing

- Taxis along Alpha and Bravo to Runway Two-Zero following ATC commands



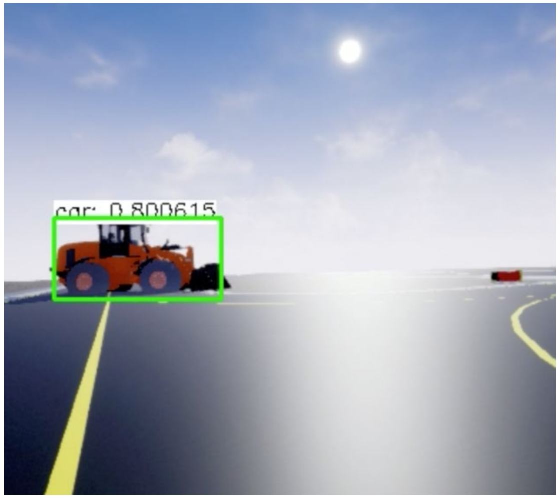




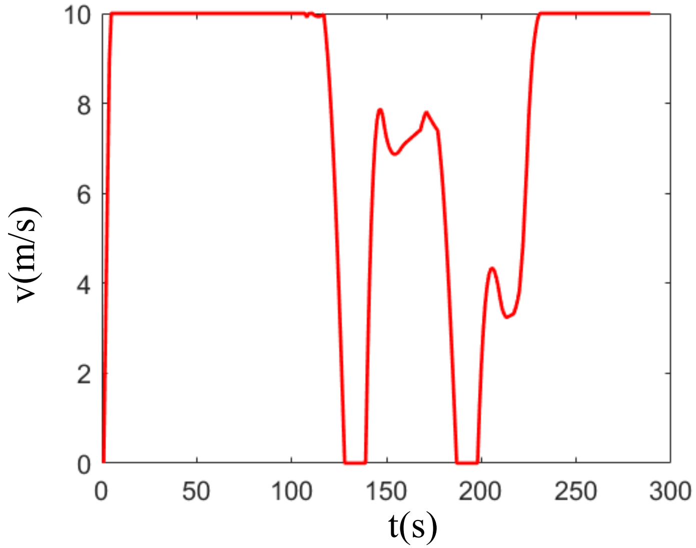
Results –Incursion Events

- Original plan: taxi along Alpha and Charlie to Runway Two-Zero
- Incursion object detected and classified as car.
- New path generated based on ATC commands.

Object Detection



Velocity Profile



Conclusions

Conclusions

Vision-guided Autonomous Taxiing:

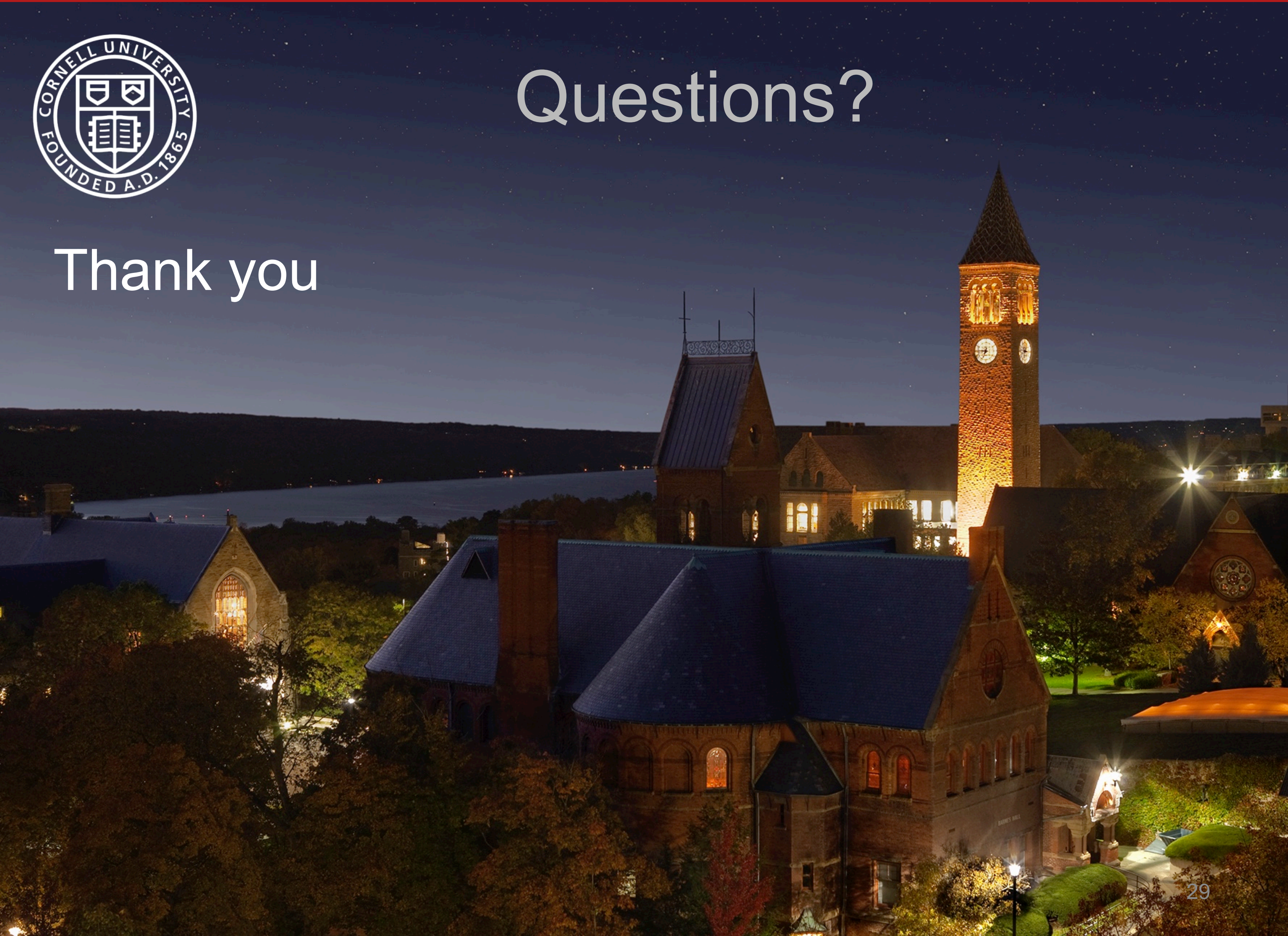
- Airport modeling using airport diagrams and geographical information.
- A systematic approach that enables real-time aircraft perception, obstacle avoidance, and feedback control with ATC commands in the loop

Future Work:

- Robustness analysis of the proposed approach
- Incorporate action recognition and environmental semantic understanding for airport environments

Questions?

Thank you



References

- Lu, W., Zhu, P., and Ferrari, S., "A hybrid-adaptive dynamic programming approach for the model-free control of nonlinear switched systems," *IEEE Transactions on Automatic Control*, Vol. 61, No. 10, 2016, pp. 3203–3208
- Zhu, P., Isaacs, J., Fu, B., and Ferrari, S., "Deep learning feature extraction for target recognition and classification in underwater sonar images," *IEEE 56th Annual Conference on Decision and Control (CDC)*, 2017, pp. 2724–2731.
- Lu, W., Zhang, G., and Ferrari, S., "A randomized hybrid system approach to coordinated robotic sensor planning," *49th IEEE Conference on Decision and Control (CDC)*, 2010, pp. 3857–3864.