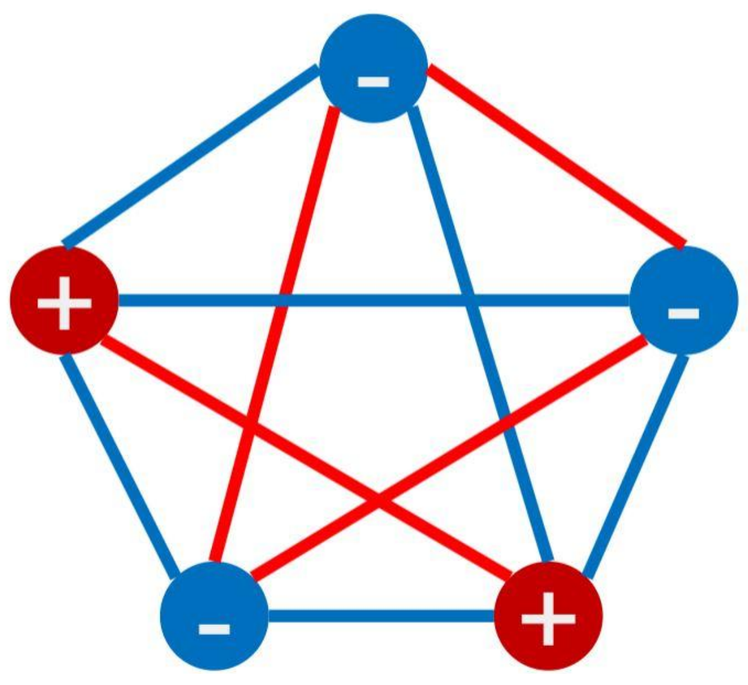
Achieving Autonomous Drone Flight via a Hopfield Network Computer Vision System

Jovan Yoshioka, Dong Jun Woun

Background & Motivation

Hopfield Network (HNN):

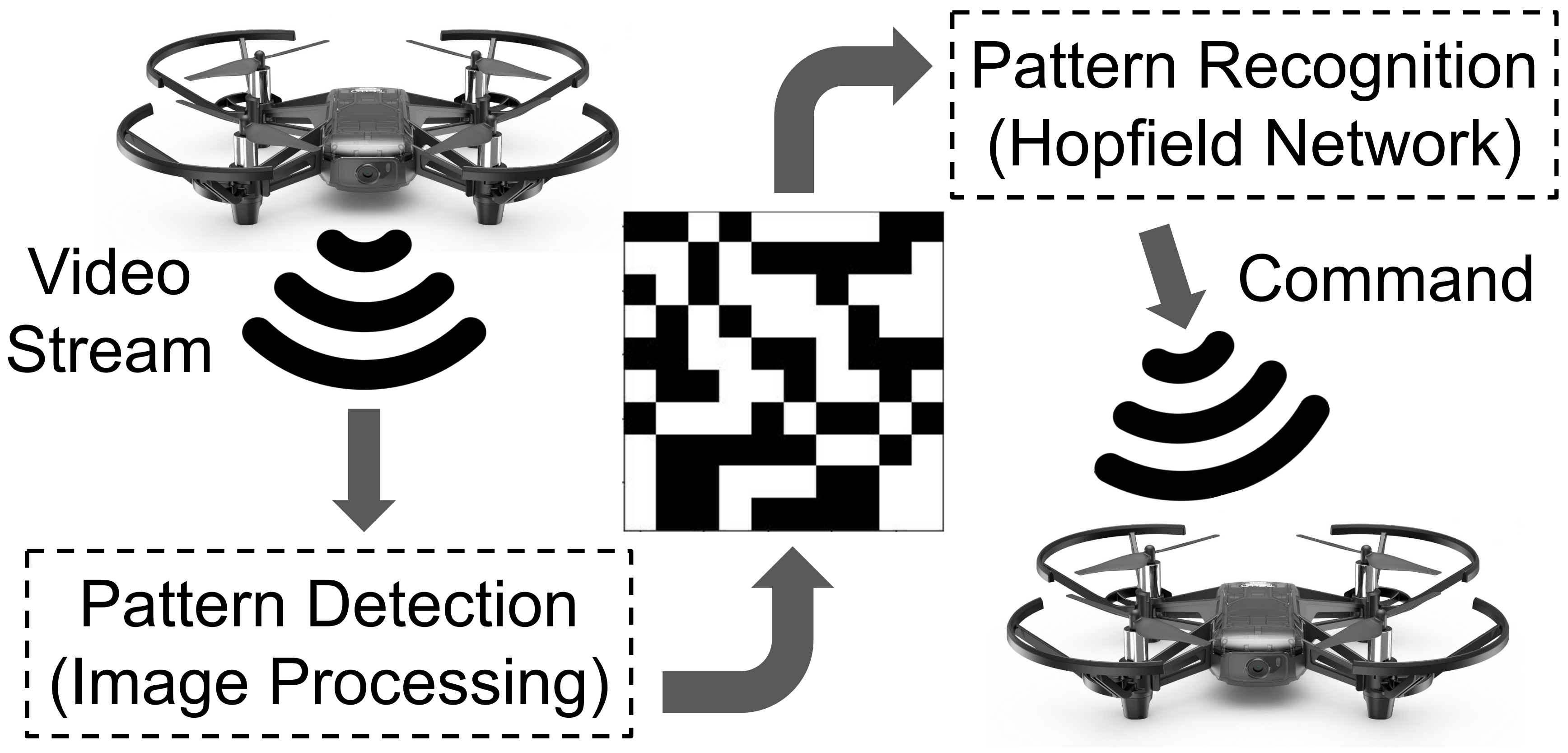
- Connected and symmetric recurrent neural network.
- All neurons impact the output with weighted connections.
- Patterns can be imprinted for optimization and associative memory.



Motivation:

- State-of-the-art computer vision models are computationally expensive, especially for real-time.
- HNNs are relatively cheaper and robust.
- Special hardware (GPUs/chips) are not needed.

Workflow

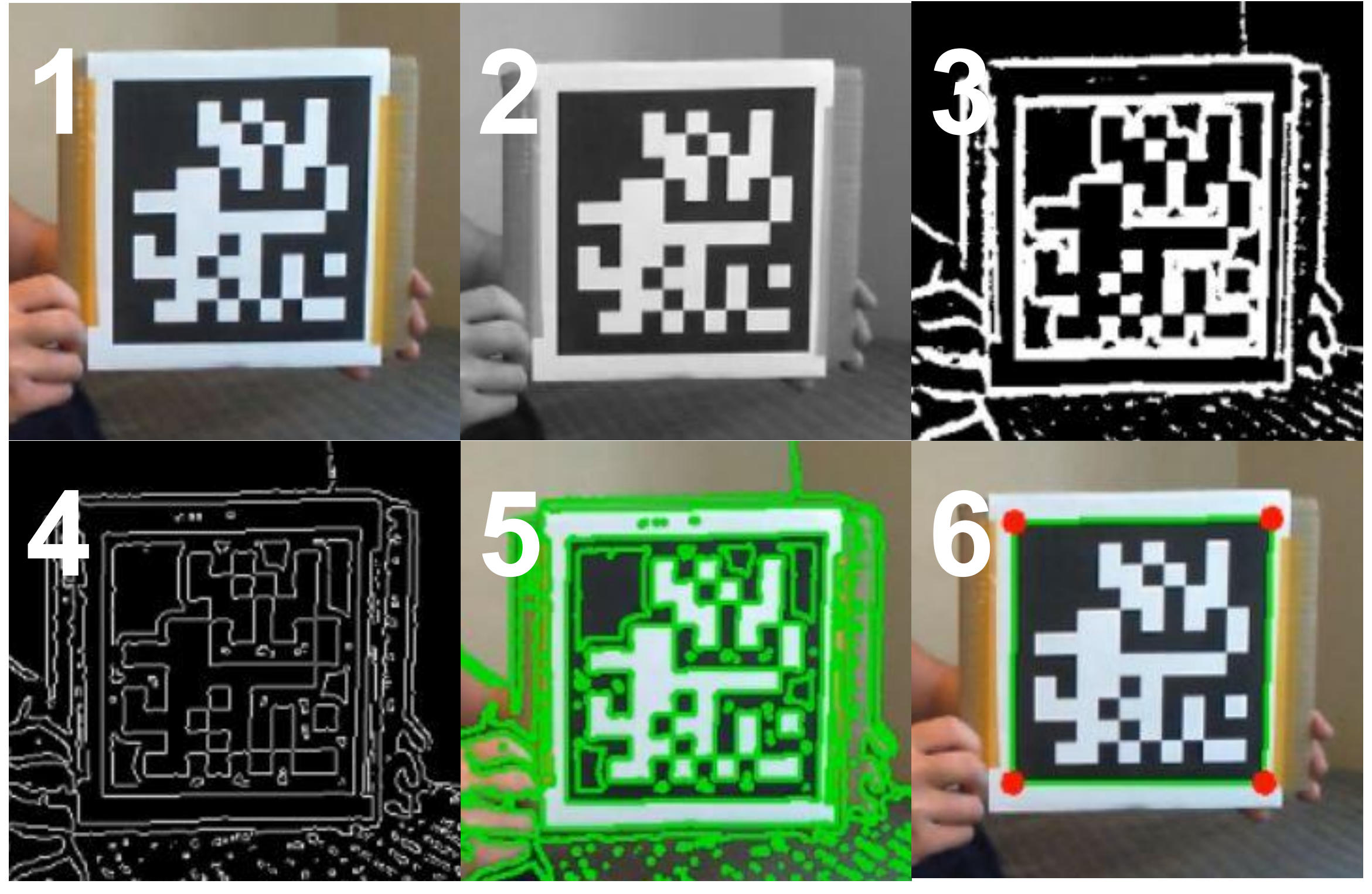

Video Stream
Pattern Detection (Image Processing)
Pattern Recognition (Hopfield Network)
Command

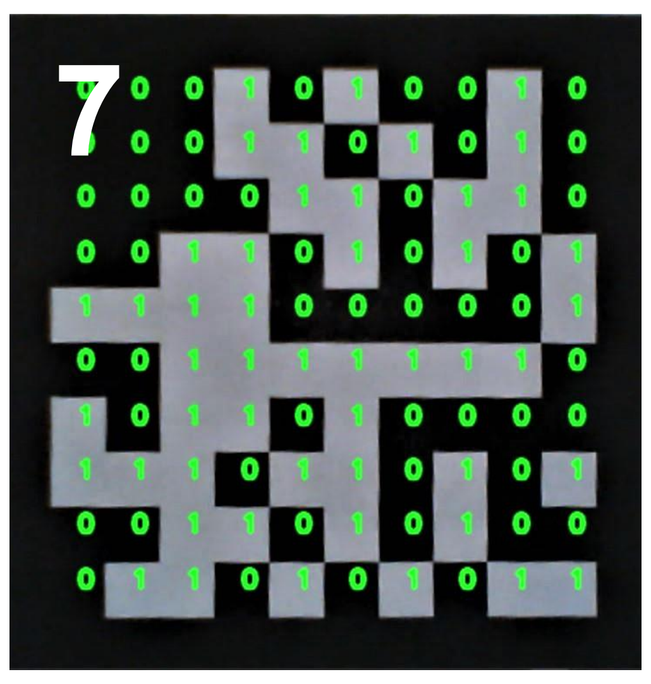
Pattern Detection (Image Processing)

- Algorithm is written in Python with OpenCV.

Detecting a Pattern:

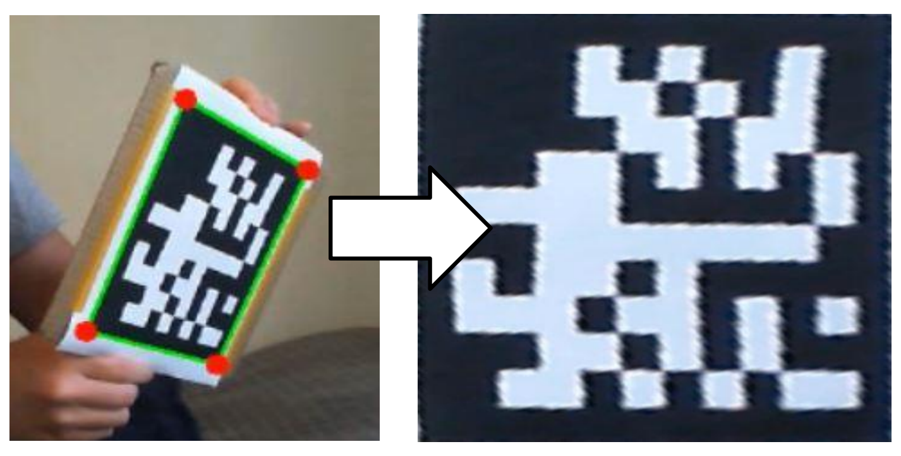
- Remove color for binary thresholding later (**Fig 2**).
- Apply adaptive threshold, ignoring lighting (**Fig 3**).
- Use Canny edge detection on the pattern (**Fig 4**).
- Find contours to make edges a points array (**Fig 5**).
- Detect pattern as a contour with four corners with several contours inside of it (**Fig 6**).
- Apply perspective transform (**Fig 8**), associate each pattern unit as 0 or 1 based on color value (**Fig 7**).



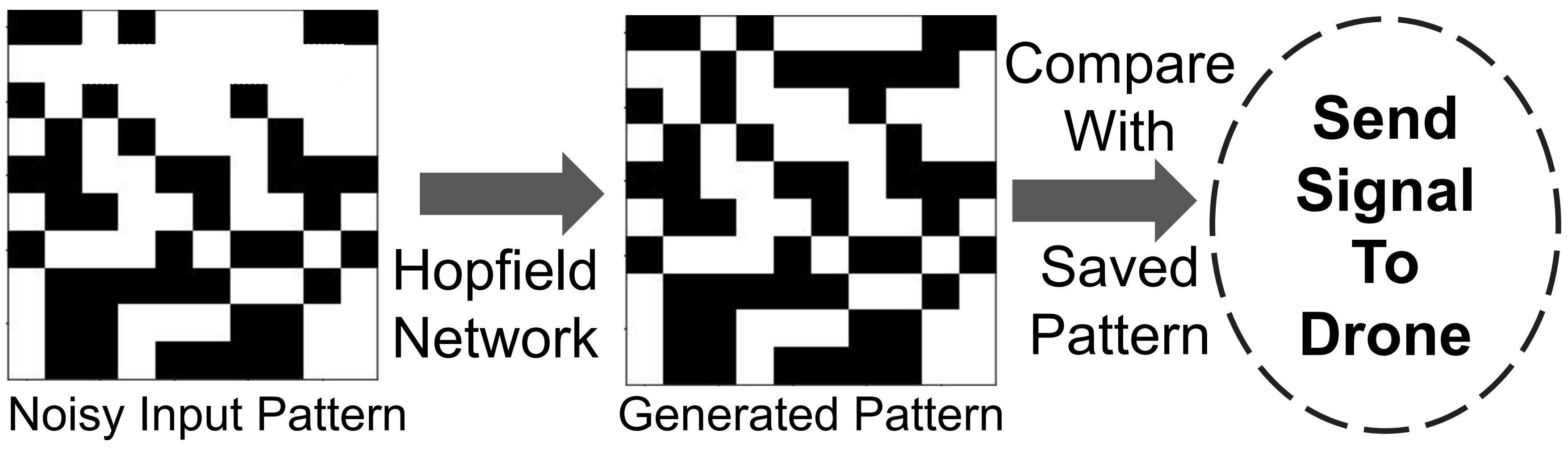


Perspective Transformation:

- Detect regardless of the pattern's orientation (**Fig 8**).



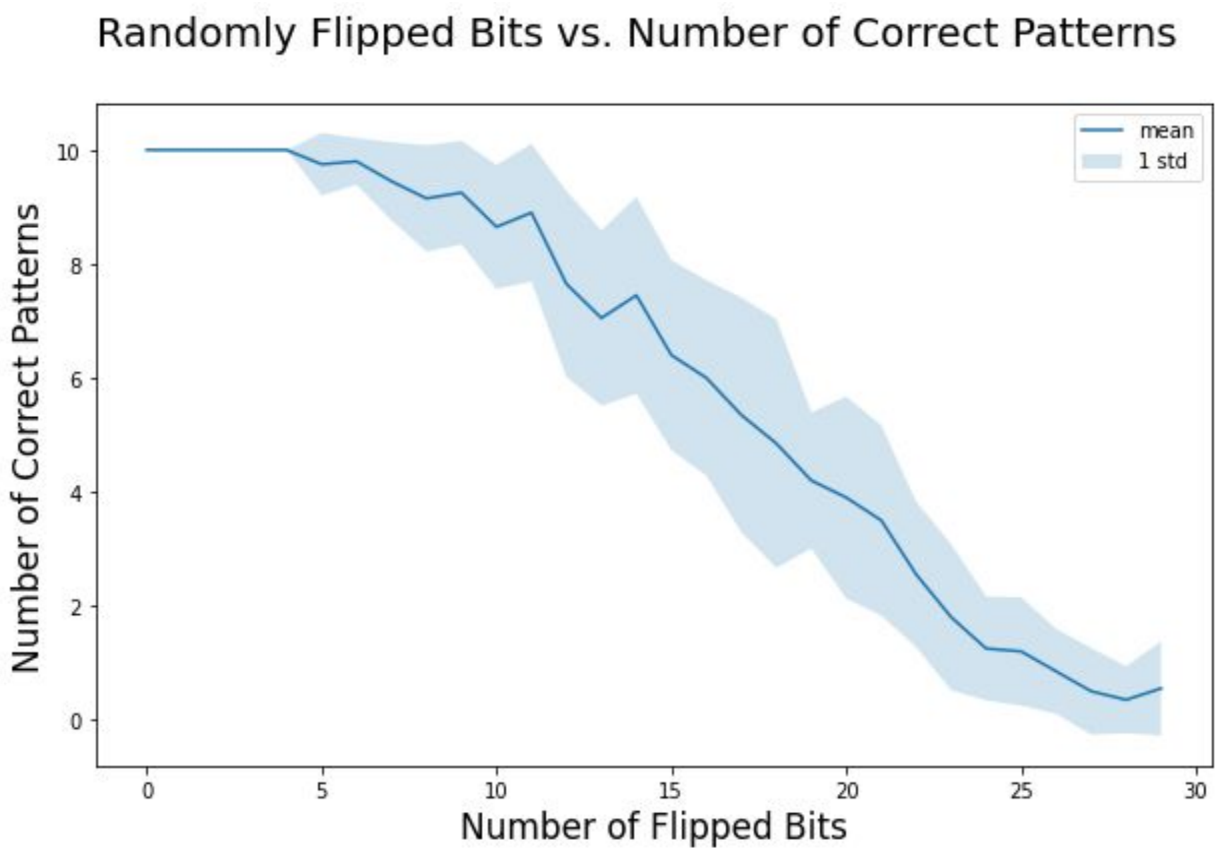
Pattern Recognition (Hopfield Network)

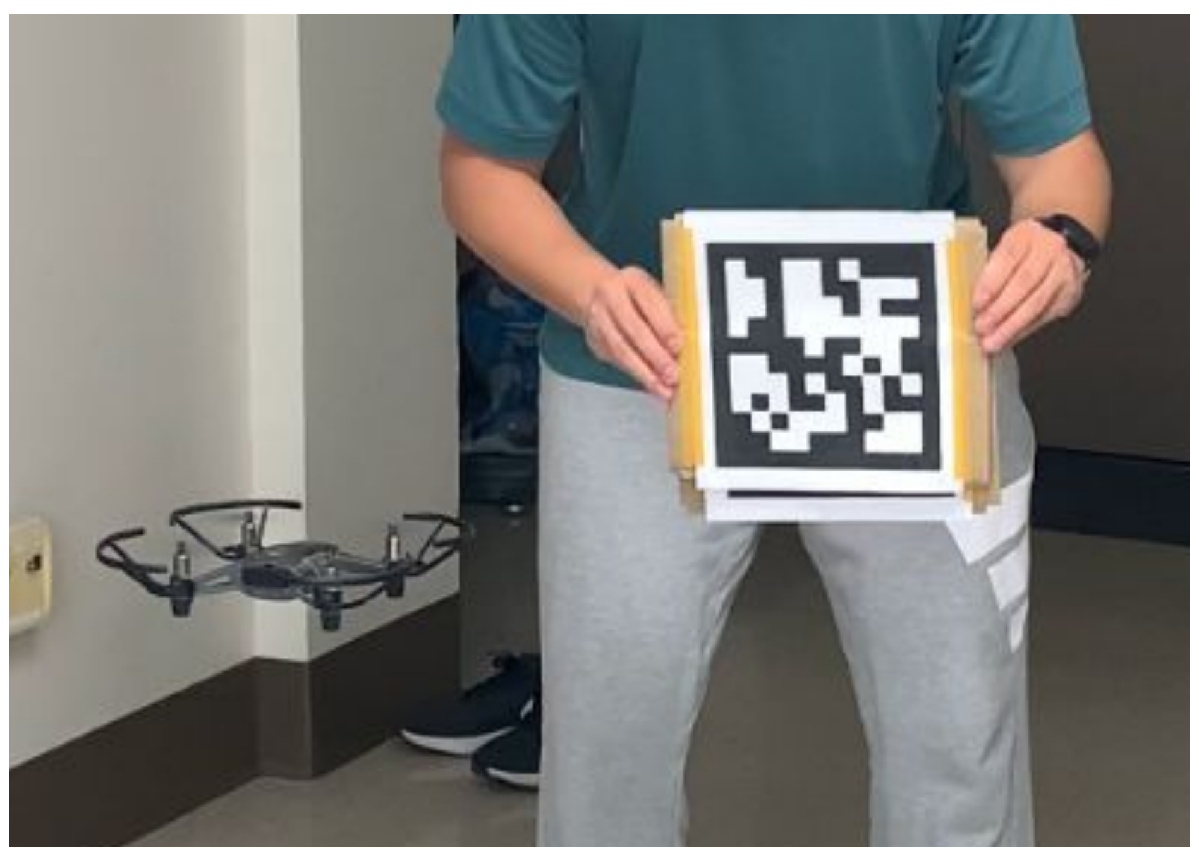


- HNN transforms the input pattern to a newly generated pattern.
- Compare generated pattern with saved patterns.
- If the generated pattern matches a saved pattern, send the associated commands to the drone.

Results

- 20 iterations with 0-30 flipped bits were tested for robustness.
- Recognized correct patterns 100% of the time with zero noise.
- Performs accurately up to 5 randomly flipped bits.





- Detects patterns from 8½ ft away.
- False detections are more likely the further the pattern.
- Works in varying lighting conditions.
- Recognizes up to 10 patterns reliably.

Conclusion

- HNN is a feasible method to implement a robust, CPU-based computer vision system for autonomous pattern-based drone flight.
- Using lightweight OpenCV filtering can achieve sufficient close range detection of patterns.

Future Work

- Expand the HNN memory capacity to increase the number of recognizable patterns.
- Improve the image processing to recognize multiple patterns at once and reuse the same pattern.
- Utilize evolutionary algorithms to optimize the number of neurons and patterns.

THE UNIVERSITY OF TENNESSEE KNOXVILLE

This project was created for the Biologically-Inspired Computation course (COSC 420) taught at The University of Tennessee Knoxville in the Min H. Kao Department of Electrical Engineering and Computer Science. Thank you to Dr. Catherine Schuman for her instruction and support throughout the course, enabling this project.