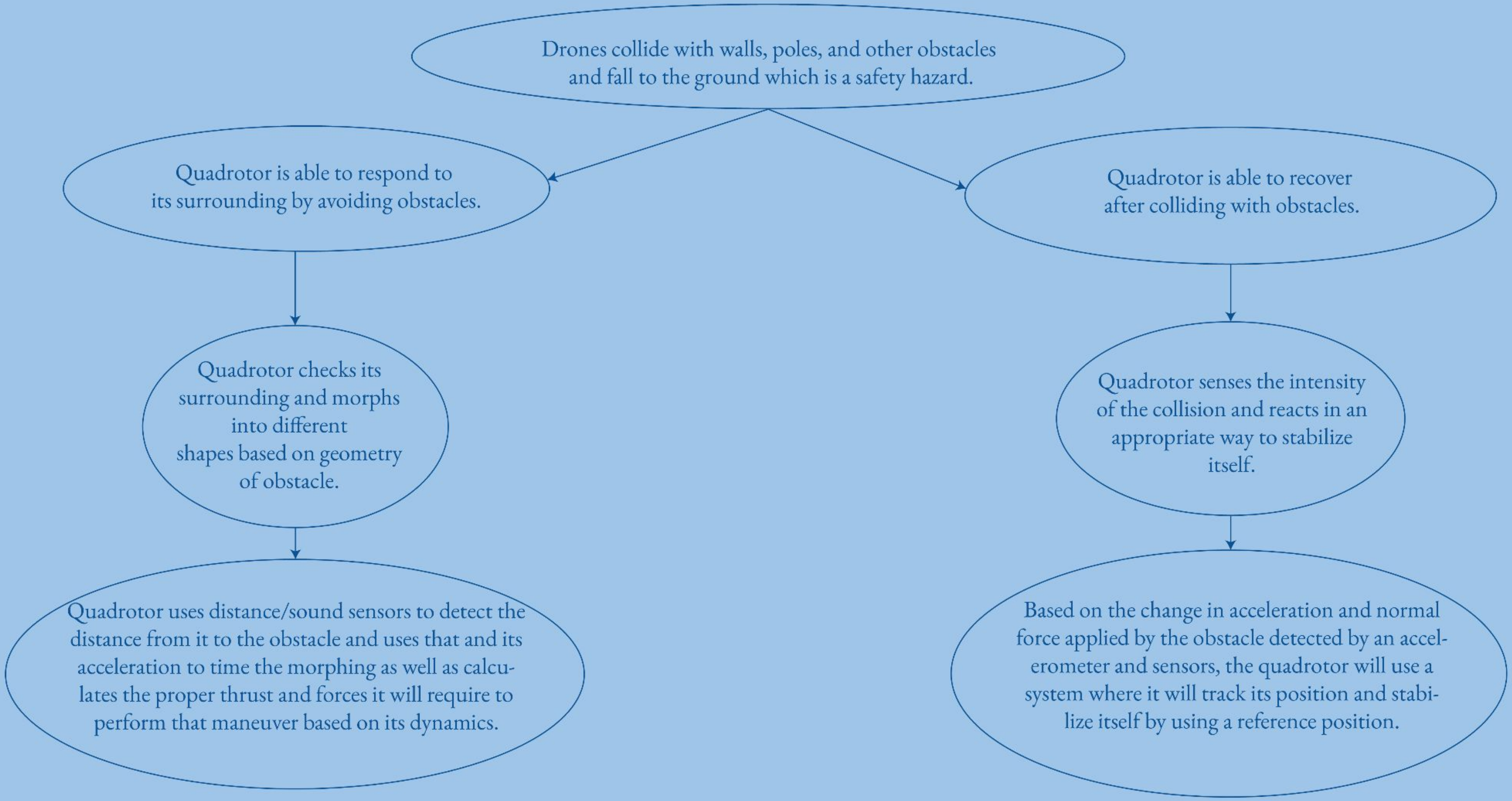
Quadrotor Collision Avoidance/Recovery

Problem Statement

Quadrotors are susceptible to colliding with its surrounding environment due to faulty piloting, unexpected wind blasts, obstacle avoidance, and many more since many commercial quadrotors are not equipped with a reactive environment-awareness system.

Duncker Diagram



Constraint Matrix

	Aesthetics	Size	Weight	Cost	Durability	Sensor Capability	Visual Range	Speed & Agility	Time of Flight	Morphability & Flexibility	Testing Environment	Hardware	Grasping & Transportability	Total
Aesthetics	-	0	0	0	0	0	0	0	0	0	0	0	0	0
Size	1	-	0.5	1	0	0	0	0.5	0.5	0	1	0.5	0	5
Weight	1	0.5	-	1	0.5	0	0	0.5	1	0	1	0	0.5	6
Cost	1	0	0	-	0	0	0	0	0	0	0	0	0	1
Durability	1	1	1	1	-	0.5	0.5	1	1	0.5	1	0.5	1	10
Sensor Capability	1	0.5	1	1	1	-	1	1	1	0.5	1	1	1	11
Visual Range	1	1	1	1	0	0.5	-	1	1	0.5	1	1	1	10
Speed & Agility	1	0.5	0.5	1	0.5	0	0.5	-	1	0	1	0	0.5	6.5
Time of Flight	1	0.5	1	1	0	0.5	0.5	1	-	0	1	1	1	8.5
Morphability & Flexibility	1	1	1	1	1	0.5	0.5	1	1	-	1	1	1	11
Testing Environment	1	0	0	0.5	0	0	0	0	0	0	-	0	0	1.5
Hardware	1	0	0.5	1	0.5	0.5	0.5	0.5	1	0.5	1	-	1	8
Grasping & Transportability	1	0.5	1	1	0	0	0	0.5	0	0.5	1	1	-	6.5

Weighting Factors

Critical Objectives	200	Morphability & Flexibility
	185	Sensor Capability
	170	Visual Range
	155	Durability
	140	Time of Flight
Significant Objectives	125	Hardware
	110	Speed & Agility
	95	Grasping & Transportability
	80	Weight
	65	Size
Non-compulsory Objectives	50	Testing Environment
	35	Cost
	20	Aesthetics

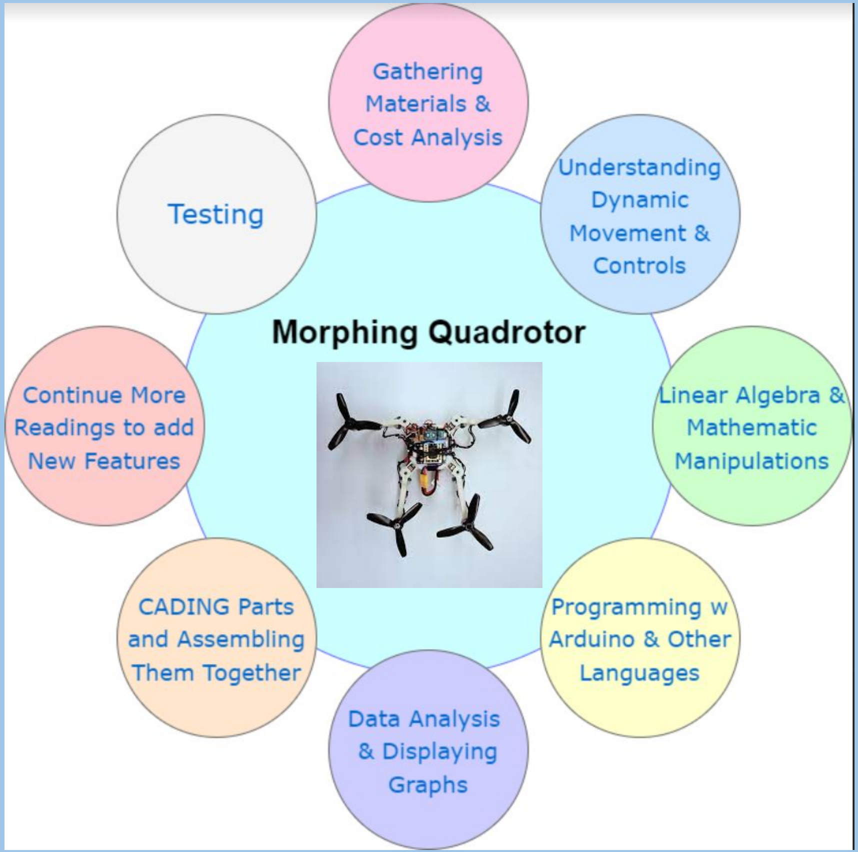
Decision Matrix

Objectives	Morphability & Flexibility	Sensor Capability	Visual Range	Durability	Time of Flight	Hardware	Speed & Agility	Grasping & Transportability	Weight	Size	Testing Environment	Cost	Aesthetics	
Weights	200	185	170	155	140	125	110	95	80	65	50	35	20	Total
Alternate Designs	Rate Value / (Rate Value × Weight)													
Morphing (ProActive)	10/2000	10/1850	8/1360	4/620	8/1120	5/625	9/990	4/380	3/240	4/260	2/100	1/35	1/20	9600
Non-morphing (Reactive)	5/1000	10/1850	8/1360	10/1550	5/700	5/625	10/1100	2/190	3/240	4/260	1/50	1/35	1/20	8350

Chosen Design: Morphing

Problem Decomposition

- Construct a budget/cost analysis alongside a constraint & decision matrix.
- A lot of parameters that need to be considered so developing this morphing quadrotor would require smaller problems to solve.
- These are some of the main tasks/problems we would have to encounter when developing this morphing quadrotor.
- Ultimately, we want to use existing knowledge of a quadrotor but make the process more efficient.



Problem Decomposition

General Procedure

- Detection → Logic → Recovery

Detection

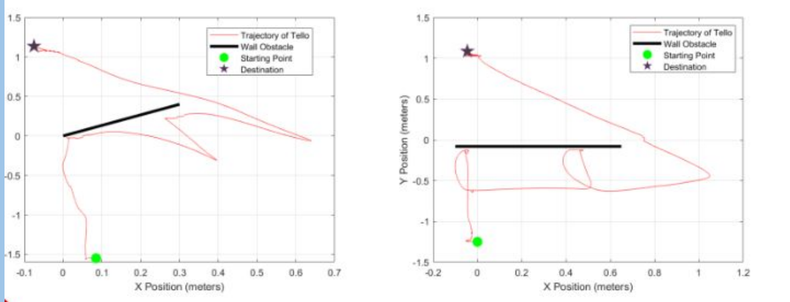
- Distance sensors
  - Sense distance from quadrotor to wall
- Accelerometer
  - Sense acceleration and calculate jerk towards wall

Logic

- Servos & PWM
  - Calculate thrust force and angular acceleration of rotors needed
  - Speed of morphing based on intensity of acceleration and jerk
- IMU Sensor
  - Measure desired angular velocity and detect motion in x-y-z plane

Recovery

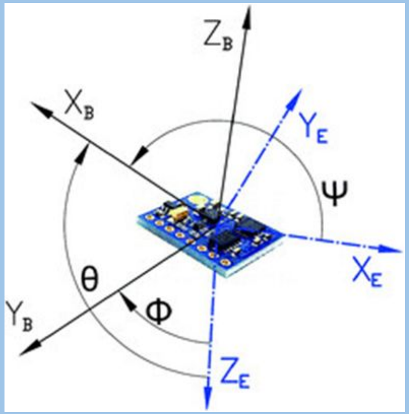
- Stabilization
  - Compute desired acceleration using intensity (preferably done by Fuzzy Logic Process)
  - Bring quadrotor to a hover
  - Stabilize altitude
- Trajectory Replanning
  - Use PID-controller
  - Use artificial potential fields (going down in PE)



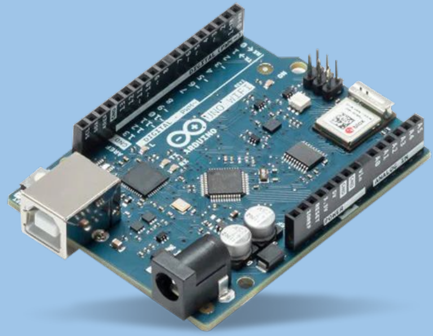
(a) 45° Wall Obstacle (b) Flat Wall Obstacle

Needed Materials

- The following are some materials we listed:
  - Quadrotor
  - IMU Sensor/Accelerometer, Distance & Sound Sensors
  - Servos (Learning how to deal with Pulse Width Modulation aka PWM)
  - Arduino Board (for starting but maybe even developing a printed circuit board to limit lots of wiring)
    - A common technique used to vary the width of the pulses in a pulse-train. PWM has many applications such as controlling servos and speed controllers, limiting the effective power of motors and LEDs.
  - Inertia & Torque Calculators
  - Bumpers (in case of collision)
    - We will be testing a lot, and sometimes some collisions can damage the drone.
  - Controller
- Cost Analysis will be created once all materials are finalized.



Including a **IMU Sensor (gyroscope)** for additional data



Skills

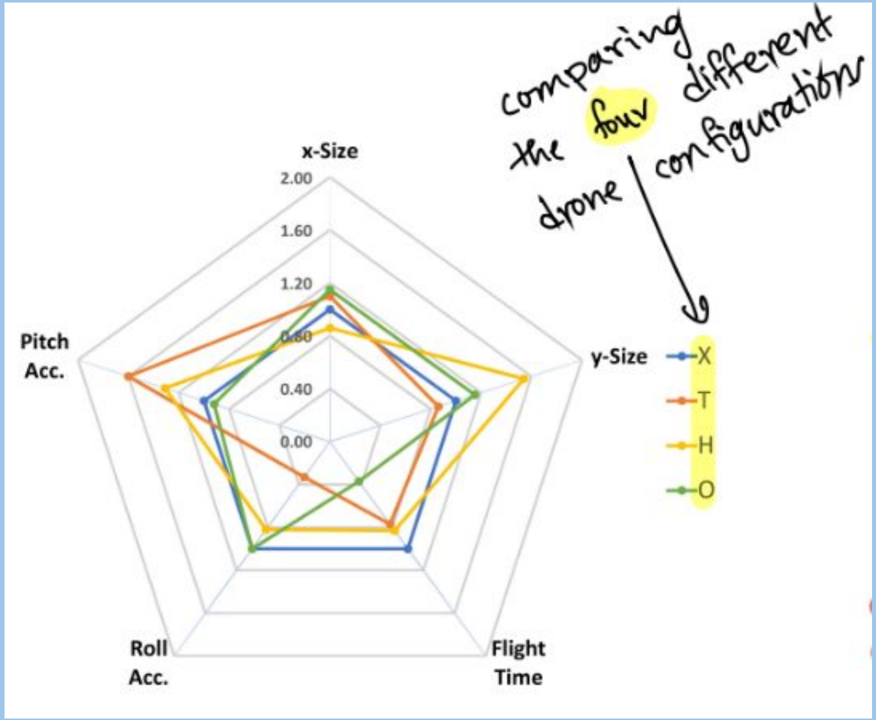
- Need to learn
  - Python
  - More MATLAB
  - **ROS**
    - CAD: CAM and FEA
- Learned
  - c++
  - CAD
  - Arduino

Important Papers

- ‘The Foldable Drone: The Morphing Quadrotor That can Squeeze and Fly’
- ‘Fly-Crash-Recover: A Sensor-based Reactive Framework for Online Collision Recovery of UAVs’
- ‘Dynamics of a Quadrotor Undergoing Impact with a Wall’
- ‘Quadrotor Collision Characterization and Recovery Control’

Paper Takeaway - 'The Foldable Drone: The Morphing Quadrotor That can Squeeze and Fly'

- 4 different configurations: X- Standard, H- narrow gaps, O- horizontal gap, T- shape proximity inspection of vertical surfaces.
- X - Standard configuration yielded the best results and was the "default" mode
- T - shape yielded best results for pitch acceleration
- H - shape yielded best results for size
- Morphing causes a drop in the hover time



Softwares

- MATLAB
- CAD (Fusion 360, SolidWorks, Inventor, Catia)
- Python
- c++
- **ROS**

Tools

- 3D Printer

ROS - Robot Operating System - Why

- Provides a standard for robotic applications
- Can be used on any robot
- Ability to connect robotic system

When

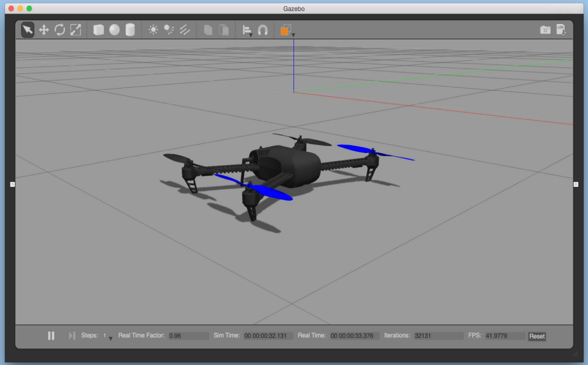
- Cramming of too many components (sensors, accelerometers, servos, motors, etc.)
- A lot of communication required within its sub-programs
- ROS is appropriate in complicated robotic systems

What

- Allows for separation of code and has communication tools (joystick, motor, camera, etc.)
- ★ Plug and Play Libraries - big time saver and has many functions like motion planning, obstacle avoidance, etc.
- ★ Language Agnostic - i.e. joystick could be coded in c++ and motor in python

Simulations

- Important to develop a simulator to predict quadrotor response to obstacle and even collisions
- Helpful for developing recovery maneuvers



Software used: MATLAB

- Can use ode45 to solve ODEs which is useful for all the mathematical equations derived in the papers
- Simulator developed in “Dynamics … Wall” paper had a 71.4% success rate with 5 out of 7 simulations matching their corresponding physical experiments
- **Monte Carlo** simulation used in “Quadrotor Collision … Control” paper
    - Used different variables to simulate quadrotor recovery under different conditions
    - In 10,000 trials, 91.9 % of the trials covered full recovery responses
    - Use of Fuzzy Logic Process (FLP) was important as it was used to determine the intensity of each collision

Important Points

- Drawbacks
    - Papers researched quadrotors colliding with small velocities
    - Paper that specialized in morphing had slow actuated joints
- Solution
    - Smaller the quadrotor, the faster the morphing
    - High speed servos (with PWM)
        - Savox 1257tg
- Adaptive Control Scheme needed
    - Takes into account geometry of morphed quadrotor and calculates geometric properties (contact geometry)

Mathematical Procedure

- All papers have very similar calculations
    - Using position vectors
        - Used to find point of contact
    - Center of Gravity
    - Inertia
        - MOI can be split into different parts (body, arm, motor, rotor, etc.)
    - Linear acceleration
    - Angular velocity
    - Desired force and thrust (normal forces)
    - Angle of inclination toward wall/obstacle
    - Generation of P.E. field and going down in P.E.
- Done in MATLAB
    - Tons of graphs as well

Next Steps

Since both of us will not be on campus for Summer 2, progression will have to be remote.

- More Readings (particularly ones are Morphing Quadrotors as we finalized that idea of a drone would be best for us) to gather more ideas regarding the dynamics and controls of the drone.
- Modifications to Matrix/Quantitative Results we need to show.
- Find a quadrotor we want to use and identify what parts to modify.
- Cost Analysis and Finalize Materials.
- CAD-ing the propellers or other parts of the drone we would like to modify.
- Different Ideas of Morphing:
    - https://rpg.ifi.uzh.ch/foldable_drone/ (We sent this link before on Teams)
    - https://video.vt.edu/media/t/1_og4ur93z


BY MORPHING BETWEEN MULTIPLE STABLE, LOAD BEARING STATES