Autonomous Rover for Mobile Support (ARMS)

By Jackson, Omri, Luke, and Hans

Overview

Introduction and Motivation

ARMS Design

Results and Demo

Future Work

Acknowledgements and Resources

References

Introduction and Motivation

Motivation

- 79 million+ students carry backpacks
- Loaded backpack should never weigh more than 15% of carrier's body weight.
- ~55% of students carry a backpack that exceeds their maximum carrying capacity
- ~23,000 backpack-related injuries in 2007

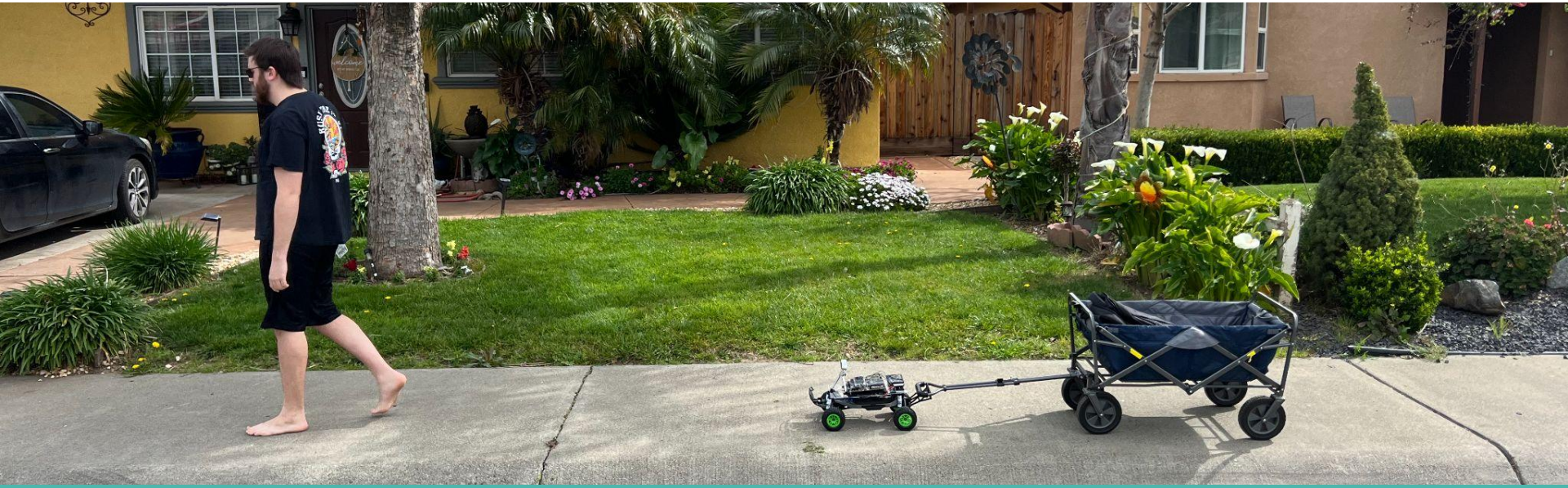
Current Solution

Autonomous rover that follows the user to carry their items.

- Piaggio's Gita implements first of its kind "cargo robot"
- The problem: The Gita gives no way for the user to interact with the robot

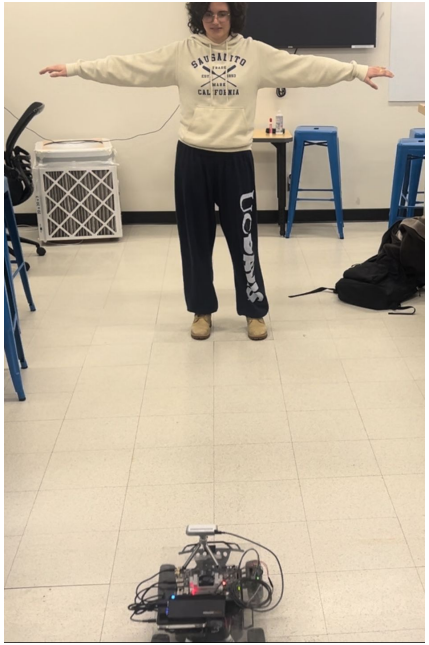
Our Solution

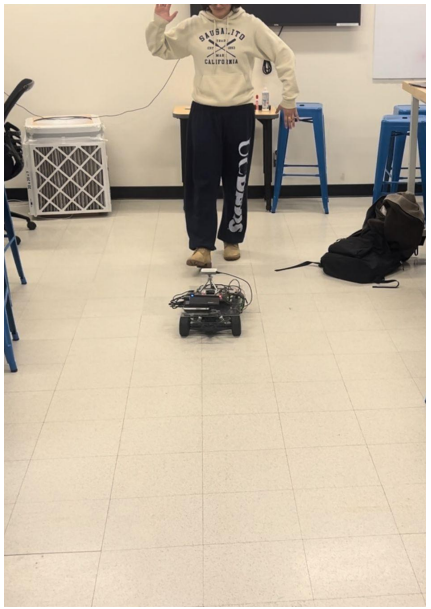
- Potential preventative measure against backpack-related injuries
- Compact rover retrofitted for cargo w/ body position and pose tracking
- Convenient use, accessible



Specifics

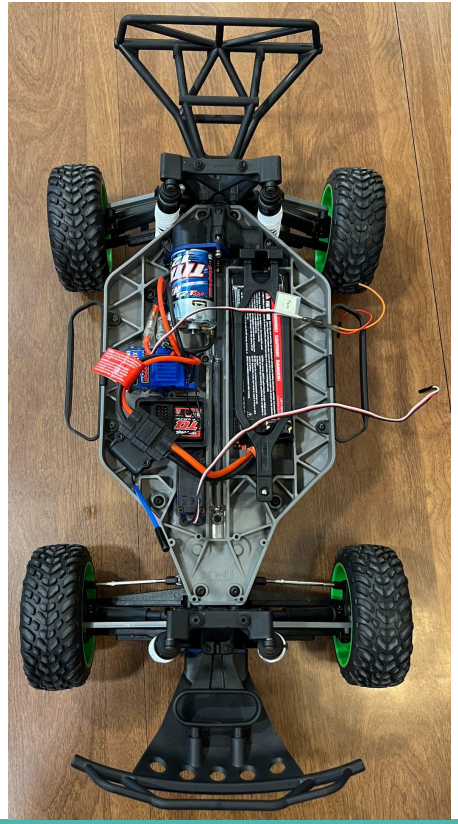
- Automated rover to carry backpacks and other cargo
  - Any chosen compatible carriage can be attached to the rover
- Controlled by posing and speed of user
  - Rover will auto brake if user leaves frame
  - Rover will auto brake if too close to user
- Turned on and off easily
  - Posing by user
- Rechargeable and swappable battery
  - Easy to use





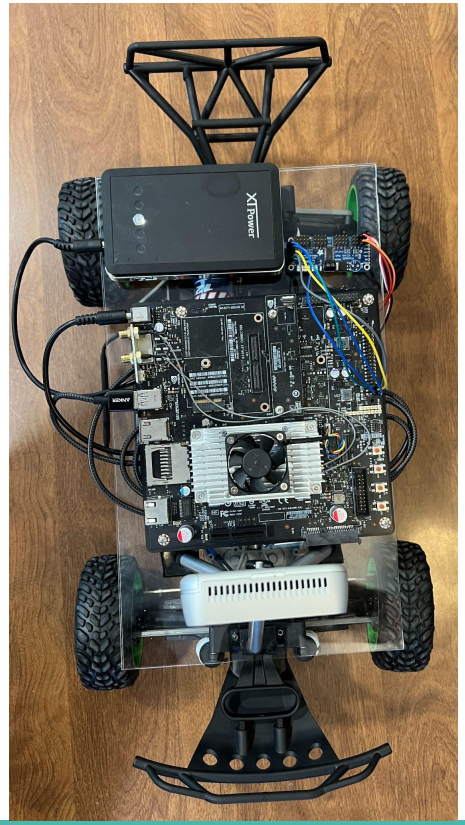
ARMS Design

Initial Hardware Setup



- Wired steering and throttle for manual use
  - Different setup than automated driving
- Calibrated the electronic speed control (ESC)
  - Specific to car type and model
- Calibrated controller to work with the ESC
- Tested functionality of the car
  - Throttle and variable speed
  - Steering
  - Braking
  - Battery life

Jetson Hardware Setup



- Wired steering and throttle for automated use
  - Controls go through I2C Servo Board
  - I2C Servo Board connected to power and commands from Jetson
- Modular back velcro for different uses
  - External battery (shown) for real world use
  - USB hub for stationary testing
- Access to Jetson
  - Through VNC for real world use
  - Through monitor, keyboard, and mouse for stationary testing

trt_pose

- Pose detection designed for Jetson
  - Easy to interface with
  - Optimized for our hardware
- Found specific points on wanted person
  - Elbow, shoulders, hands, head, and chest
- Explored functionality of trt_pose
  - Poses
  - Tracking
  - Sorting

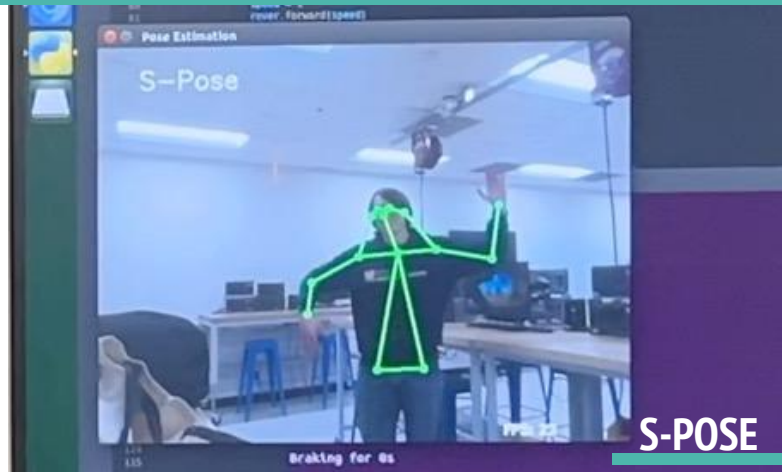
Model	Jetson Nano FPS	Jetson Xavier FPS
resnet18	22	251
densenet18	12	101

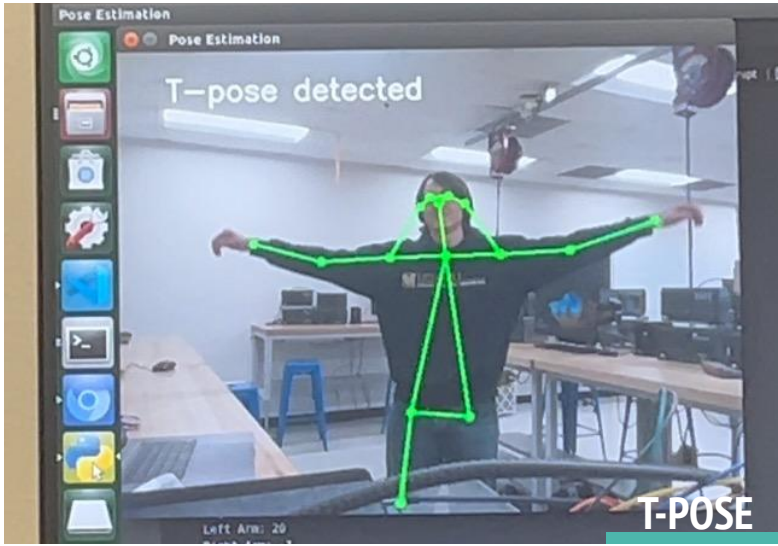
Chest-Based Tracking

- Follow chest point on the user
  - Calculate horizontal distance from center of camera feed to the chest point
  - Adjust wheel angle based on the difference
    - Scaled to the max and min turn radius
- Braking based on chest point
  - Rover will stop if:
    - Chest point leaves frame
    - Chest point is too close to camera
  - Rover will only go if:
    - Chest point is in frame
    - Chest point is further than specified depth

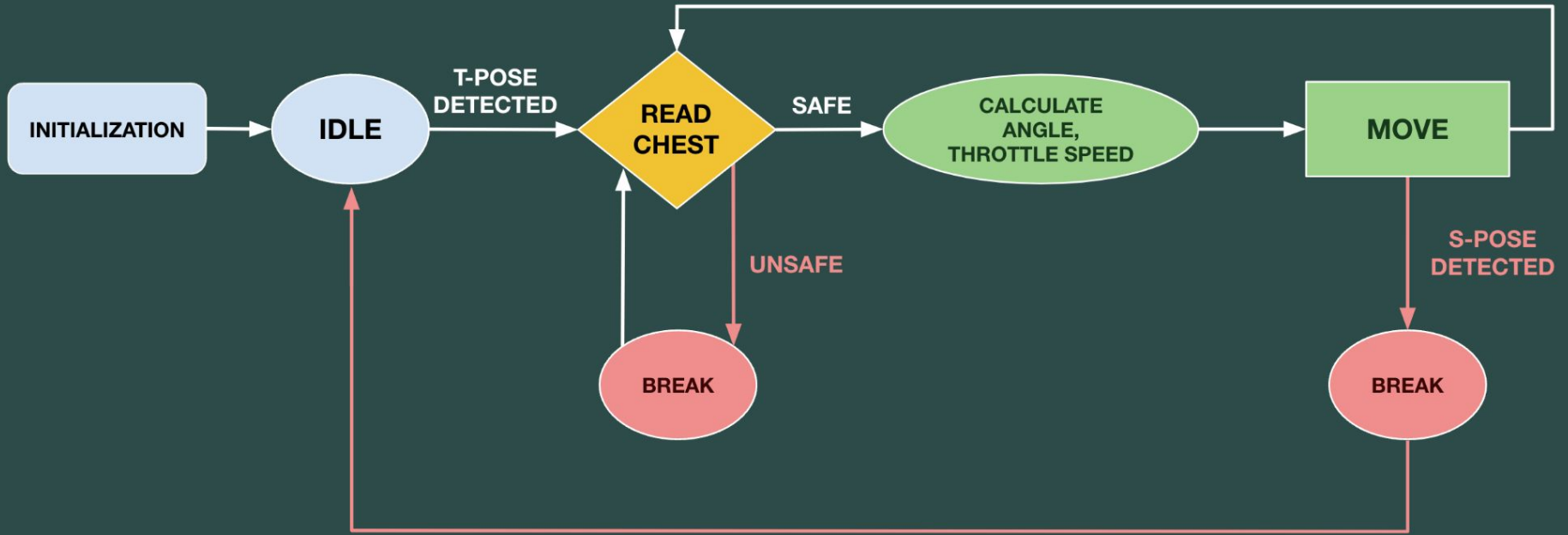
Pose-Based Control

- Calculate poses with body points
  - Comparison to each other
  - User chest must be in frame
  - User chest must be at valid distance
- T-Pose starts the car
  - Hands, elbows, and shoulders in a row
- S-Pose stops the car
  - Body points in a sideways S form
- Old poses not used
  - W-Pose to speed up
    - Dangerous!!!
  - M-Pose to slow down
  - Distance measuring used instead





Automated Pipeline

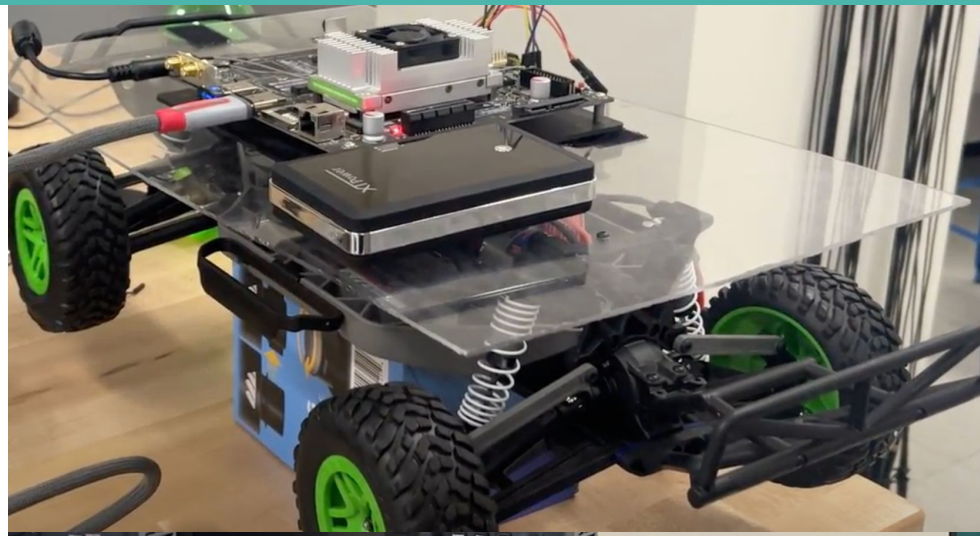


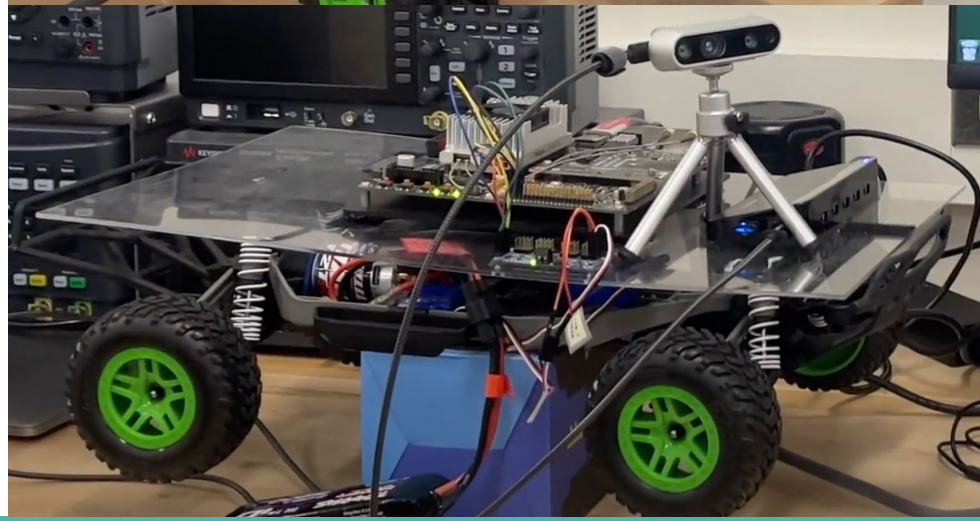
Remote Access

- Connect to Jetson virtually to run rover from laptop or phone
  - Easier than having to carry around screen, mouse, and keyboard
  - We used the X11 VNC
    - Linux-friendly
- Eduroam complications
  - Eduroam security did not allow rover to connect
  - Phone hotspot was used for laptop and rover
  - Easy to attach to rover and have amazing service

The Old Acrylic

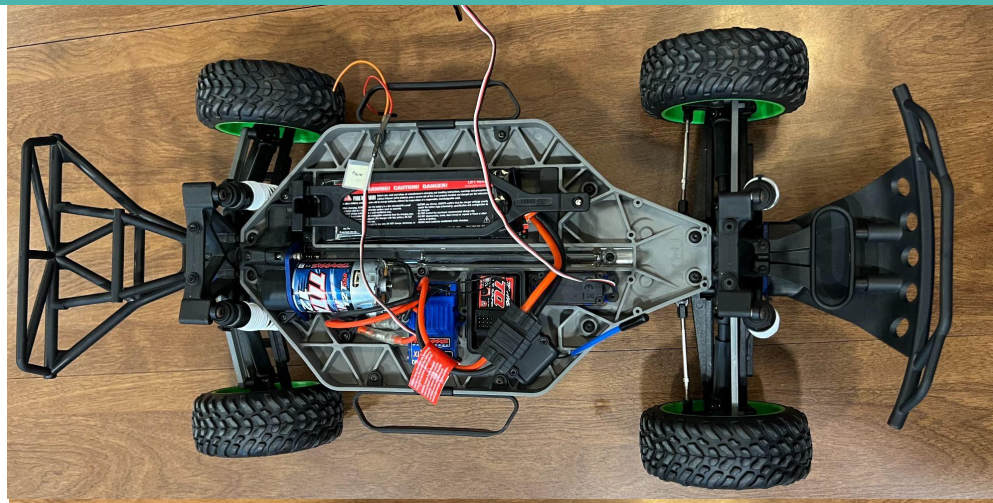
- Acrylic was larger than the rover
  - Built-in collision absorption nullified
- More acrylic surface area than needed
  - Wasted space
  - Uneven distribution of weight
- Too much useless weight on car
  - Shocks needed to be upgraded
    - Tripled shock strength
- Improperly secured
  - Moved around during use
    - Effected camera functionality

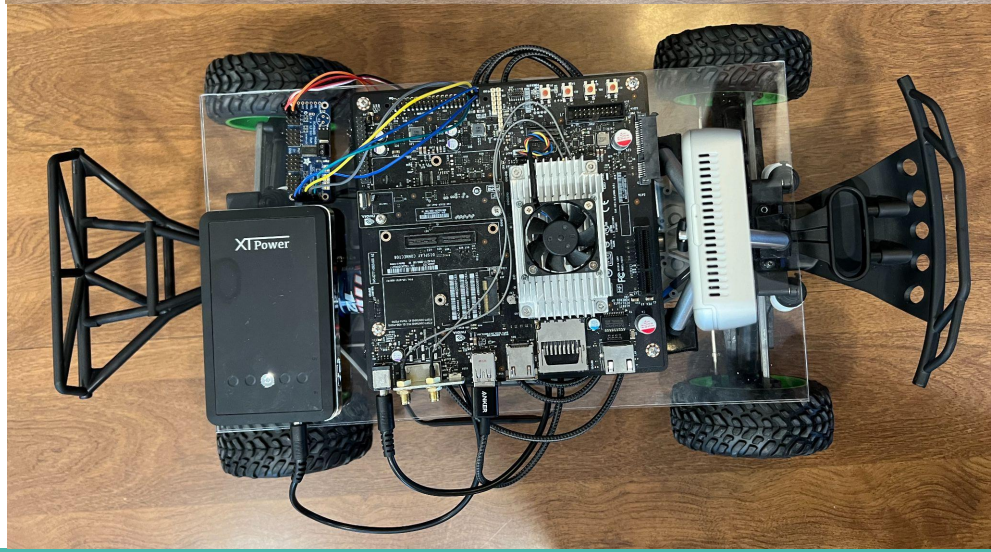




An Improved Acrylic

- Smaller acrylic to utilize built-in shock absorption
- Better use of area on acrylic
- More modular velcro
  - Battery in back for ground use
  - USB hub in back for desk use
- Lower amount of weight on car
  - Weight was also more centered





Results & Demo

Final Product

- Follows user well
- Pose detection functions
  - Fast on Jetson
  - Begins and ends properly
- Automatic braking
  - Too close to an object
  - No person is in frame
- Average frame rate of ~20fps

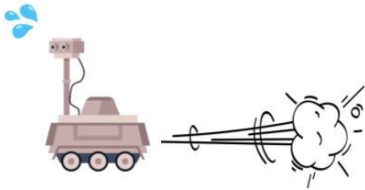
Future Work

Tuning Current Implementation

- Rover had very sensitive movement
  - Better communication between processes
  - Tuning
  - More solid construction
- User needed to be vigilant of sharp turns
  - Camera view change
    - Higher
    - Wider
  - Code optimization
  - Closer following
- Rover confused person-like objects as people
  - Tuning
  - MOTS

Variable Following Distance

- Currently following distance is hard-coded into the program
  - Make a pose to increase or decrease following distance
- Rover can then automatically change speed depending on distance
  - Higher speed if too far
  - Lower speed if too close
- PID controller functionality for proper distance
  - Continuously variable speed



Multi-Object Tracking & Segmentation (MOTS)

- Currently works with only one person in frame
  - Only single person functionality
  - TRTPose may have built in tracking
- Rover would be more functional in areas with many people
- Would work well with the variable distance
- Camera position would ideally be moved up for both features

Acknowledgements and Resources

Acknowledgements

We extend our gratitude to Professor Chuah for her invaluable guidance throughout this project. We also wish to thank Kartik Patwari for his technical support and the ECE Department of The University of California, Davis, for providing the necessary funding. Their collective support was instrumental in the realization of A.R.M.S.

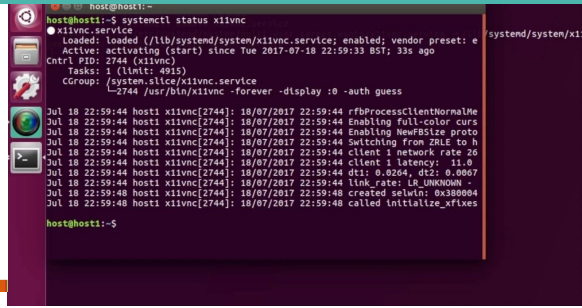
Resources

- Hardware
  - Traxxas Ford Fiesta
  - NVIDIA Jetson TX2
  - Intel RealSense
- Software
  - trt_pose
  - X11VNC
  - CV2

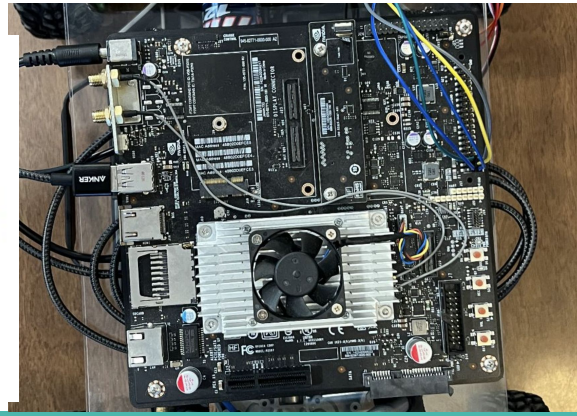
NVIDIA-AI-IOT/
trt_pose

Real-time pose estimation accelerated with NVIDIA TensorRT

2 Contributors · 129 Issues · 932 Stars · 290 Forks



OpenCV



References

References

- “Real-Time Human Pose Estimation.” *NVIDIA Developer*, developer.nvidia.com/embedded/community/jetson-projects/nv_trt_pose. Accessed 9 Mar. 2024.
- Nvidia-Ai-Iot. “Nvidia-Ai-IOT/TRT_POSE: Real-Time Pose Estimation Accelerated with Nvidia TENSORRT.” *GitHub*, github.com/NVIDIA-AI-IOT/trt_pose. Accessed 9 Mar. 2024.
- “Jetsonhacks - Overview.” *GitHub*, github.com/jetsonhacks. Accessed 9 Mar. 2024.
- @sabelskaya, “Lunar rover or moonwalker space station icon, flat vector illustration isolated. stock illustration”, iStock. Accessed 14 March 2024.