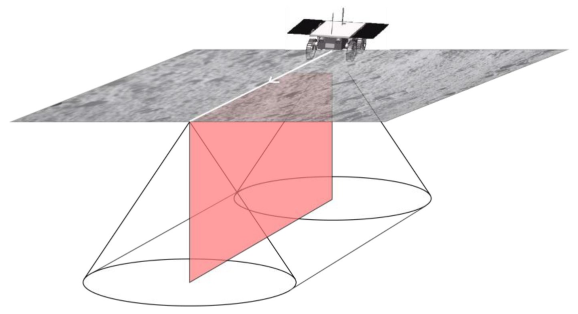
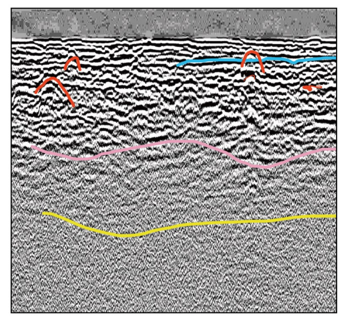
# Machine Learning for Lunar Multi-Static Ground Penetrating Radar

**STUDENTS:** Bhanu Sharma, Cheng-Hao Hsiao, Chi Chen, Jyothirmayee Donthineni, Meharleen Singh, Piyush Tandon, Samuel Richard Kinn

## Concept

- There are many applications where it is desirable to image the subsurface nondestructively. In this case, Mars and the Moon
- Ground-penetrating radar (GPR) provides a means of doing this in a cost-effective manner
- Our goal was to deploy an image processing and machine learning pipeline to extract meaningful information about the surface from GPR data
- This pipeline could then be used in future research and development on this project, including in future ENGINE projects.

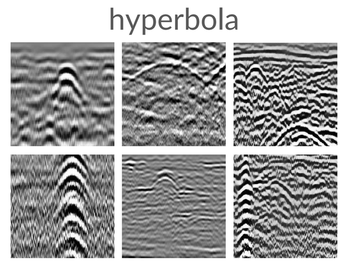




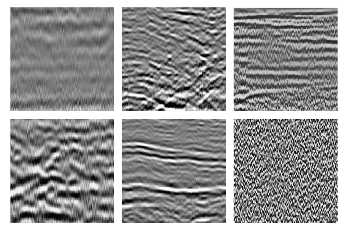
## GPR image and Target

- GPR devices send radar pulses into the ground and use antenna to capture signals that have returned as pulse is reflected, refracted, scattered back to the surface
- Strongest signal strength of antennas occurs close to the ground
- EM properties of materials can limit penetration depth of radar
- From these multidimensional signals, 2D and 3D projections can be created and represented as images

hyperbola



not hyperbola



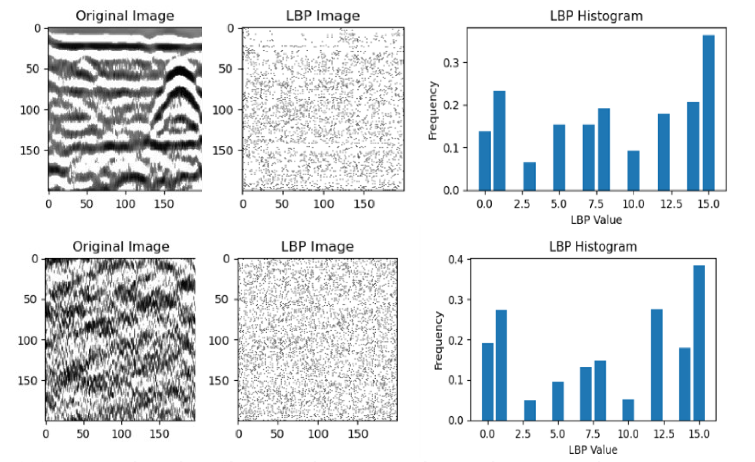
## Feature extraction

- Preprocessing techniques, such as contrast enhancement, noise reduction, and image denoising can help to accentuate the discriminators or identifiers of interest, making them more distinguishable for subsequent analysis.

### Features for hyperbola classification

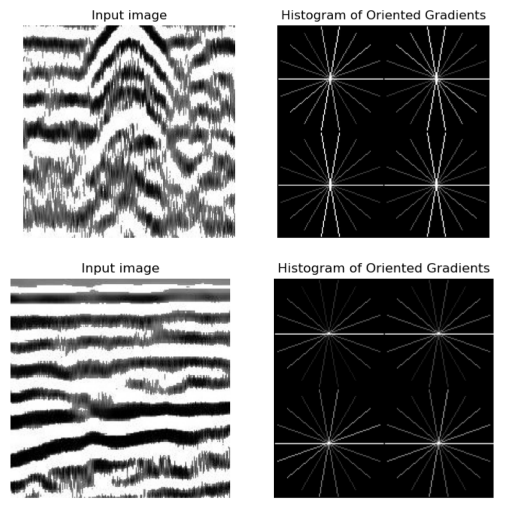
- In traditional machine learning, features play a crucial role in the learning process and the subsequent performance of the model. Features allow domain experts to incorporate their knowledge into the model and can provide interpretability and explainability to the model's predictions.

1. Local Binary Pattern Histogram
   - distinguishing different textures




Different lbp distribution between hyperbolas and environment

2. Histogram oriented Gradient :
   - analyzing the local gradient patterns



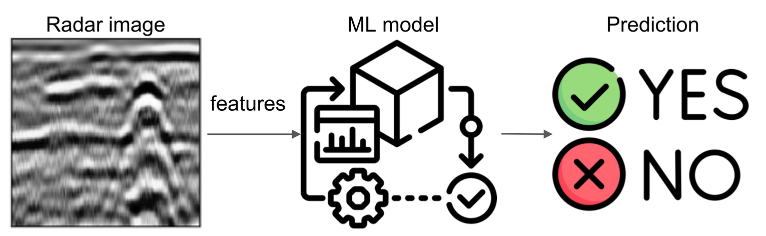

## ML Architecture

**Feature selection for model training**

Using RFE (recursive feature elimination) to get the dominant features:

1. LBP histogram
2. Fourier shape features
3. Histogram oriented gradient
4. GLCM features

Using ML model to predict the hyperbola




### Model selection and training

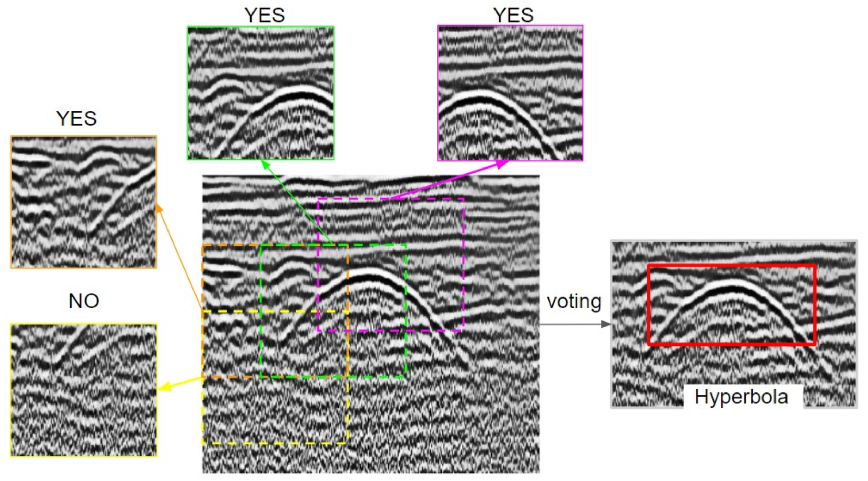
- Using decision trees, random forest, and support vector machine to train the hyperbola classification model with listed dominant features.
- Using grid search to determine the hyper parameters and optimize the accuracy of models.

Evaluating the performance of models

```
Classification Training Report:
              precision    recall  f1-score   support

           0       0.93      0.98      0.95      1615
           1       0.85      0.61      0.71       329

    accuracy                           0.92      1944
   macro avg       0.89      0.80      0.83      1944
weighted avg       0.91      0.92      0.91      1944

Train Confusion Matrix:
 [[1579   36]
 [ 127  202]]

Train Accuracy:  0.9161522633744856
```

### Hyperbola detection in radar image

- Using sliding window to scan the radar image box by box and use ML model to predict.
- Analyzing the probability of the area containing hyperbola.




Determining the hyperbola region (redbox) by the voting result of multiple predictions.

## Hardware Interface

Here are **several** compelling **reasons** why we chose using the **Raspberry Pi**:

- **Affordability:** The Raspberry Pi offers an affordable computing solution, making it accessible to a wide range of users. Its low cost enables experimentation, learning, and prototyping without breaking the bank.
- **Flexibility:** The Raspberry Pi's flexibility is one of its key strengths. It can be used for various purposes – opening up potential to flash it with a custom kernel and to get the best testing and performance results.
- **Community and Support:** The Raspberry Pi has a vibrant and supportive community. With millions of users worldwide, there is a wealth of online resources, forums, and projects available – making debugging super easy saving lots of time.
- **Educational Value:** The Raspberry Pi was originally designed with educational purposes in mind. It provides a hands-on platform for students and learners to understand computer science, programming, and electronics. Its user-friendly interface and extensive educational resources make it an excellent tool for teaching STEM subjects.
- **Operating System Compatibility:** The Raspberry Pi supports a wide range of operating systems, including Linux distributions (such as Raspbian, Ubuntu, and others), allowing users to choose the one that best suits their needs and expertise.
- **Energy Efficiency:** The Raspberry Pi is known for its energy efficiency, consuming very little power compared to traditional computers whilst still delivering competitive performance making it a perfect fit for this GPR project.

## Hardware Implementation for ML

**How we leveraged the GPU on Raspberry Pi to accelerate ML tasks:**

### 1 Pre-checks

First we need to start of with a operating system that supports hardware acceleration. Then we can Install the Mesa V3D graphics driver, which provides hardware acceleration for the GPU. To enable OpenGL ES support, install the libgles2-mesa package. Utilize the OpenMAX IL API for multimedia acceleration by installing the libomxil-bellagio package.

### 2 Compile Libraries

To take advantage of GPU acceleration, you need to compile libraries with GPU support. For example, for machine learning tasks, we compiled TensorFlow with GPU support using the TensorFlow source code and the appropriate dependencies.

### 3 Check Functionality

After Modifying the Raspberry Pi's boot configuration to enable GPU support. Allocate a specific amount of RAM for GPU memory and enable the GPU driver. To ensure proper GPU functionality, we executed some common test programs – to verify we were getting full performance. We ran OpenGL or OpenCL programs to verify GPU rendering and parallel computing capabilities.

### 4 Implemention

By following these specifications and details, we were able to unlock the power of GPU acceleration on Raspberry Pi. It enabled us to leverage the GPU's capabilities for enhanced performance and efficiency in various computational tasks – such as efficiency in real time inference using our trained models or maximizing the framerate for live GPR data being processed.

Computer Vision Libraries:

- OpenCV
- scikit-image
- Pillows
- PyTorch

Machine Learning Framework:

- scikit-learn
- TensorFlow
- Keras

## Future Work, References, and Acknowledgments

- Further improvements to SVM model
- Labeling even more features that could potentially be of interest
- Upgrade hardware to Snapdragon development board

Faculty Mentors: Payman Arabshahi, Jenq-Neng Hwang
Teaching Assistants: Harsha Vardhan, Kelly Ho, Shruti Mishra

[1] A. G. De G. Matthews, M. Van Der Wilk, T. Nickson, K. Fujii, A. Boukouvalas, P. Le´on-Villagr´a, Z. Ghahramani, J. Hensman, GPflow: A Gaussian process library using TensorFlow, The Journal of Machine Learning Research 18 (1) (2017) 1299–1304.

ELECTRICAL & COMPUTER ENGINEERING
UNIVERSITY of WASHINGTON

**ADVISERS:** Daniel Nunes, Jack Lightholder, Jonathan Sauder

**SPONSORS:** NASA JPL