Team Name: Orbital Outliers
Name of College(s)/University(s): Academy of Technology

Team Members Details:

1. AMITAYU GHOSH
2. AKASHDIP MAHAPATRA
3. SUBHASISH DE
4. DHRUBA MONDAL

Detailed solution and Approach (250-300 words)

AI/ML model development for automatic detection of craters and boulders will be developed using a fine-grained approach with Orbiter High Resolution Camera images through image processing and deep learning. The approaches described herein have the key steps as follows:

Image pre-processing: A step of OHRC image pre-processing will selenoreference images using the provided side information. This step will correct geometric distortions and registration with respect to a common Lunar reference frame.

Model Training: One can use a Convolutional Neural Network architecture, such as U-Net, which is designed for image segmentation tasks. It will be trained based on labeled datasets whereby craters and boulders are annotated. In this case, the model will learn features corresponding to craters and boulders of different shapes and dimensions.

Data Augmentation: Robustify the model by applying the following data augmentation techniques: rotation, scaling, and contrast adjustment to simulate other illumination conditions and views.

Detection and Quantification: The locations and boundaries of craters and boulders will then be predicted in new OHRC images by the trained model. Apply post-processing techniques in refining these predictions in order to filter out false positives.

Selenographic Position and Dimension Calculation: Implement an algorithm that will calculate the selenographic coordinates and diameters of the detected craters and boulders. This includes converting the pixel coordinates from the segmented images into lunar coordinates.

Shape File Generation: Generate a polygonal shape file or any other form of geospatial data format containing the boundaries and associated attributes, namely size and position, of the detected craters and boulders.

Evaluation: By evaluating the model on test cases, check the accuracy with precision, recall, and F1-scores by comparing the features it detected against a manually labeled validation set.

Tools and Technology Used (50 words)

Programming Languages: Python, C, C++

Libraries: TensorFlow/PyTorch (for deep learning), OpenCV (for image processing), GDAL (for geospatial data handling), Keras and Shapely library for creating and manipulating shapefiles, Matplotlib and Seaborn for data visualization and model evaluation

Software: QGIS (for shape file handling and visualization)

Data Formats: Polygonal shapefiles for crater/boulder boundaries

Image Processing Techniques: Edge detection, morphological operations

Opportunity should be able to explain the following:

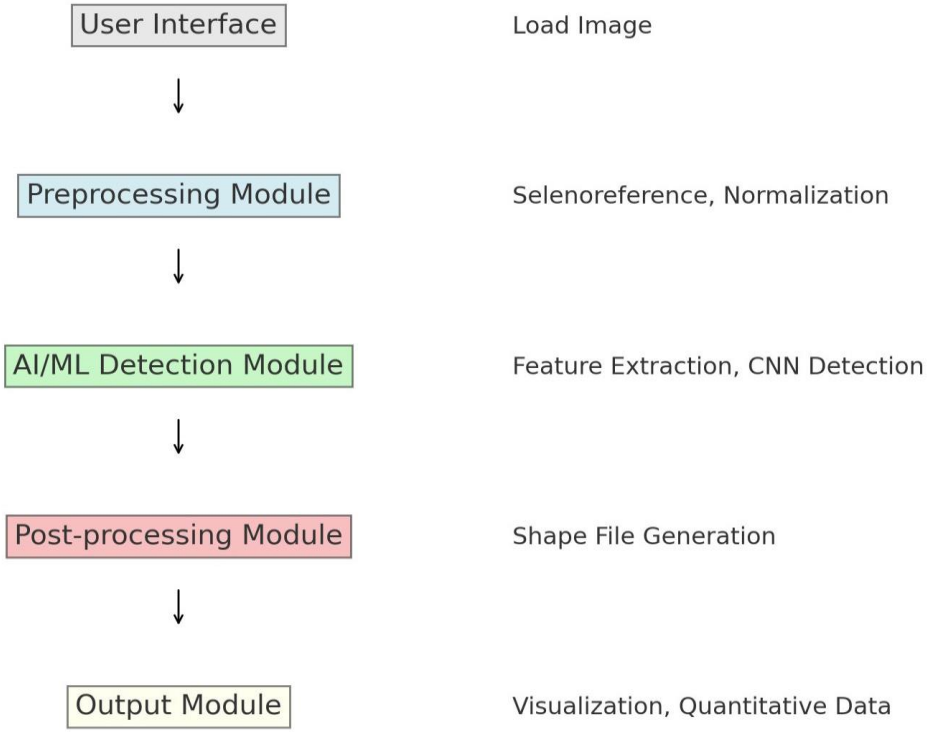
- How different is it from any of the other existing ideas?
- How will it be able to solve the problem?
- USP of the proposed solution

Most of the existing methods for crater detection are based on conventional image processing techniques that have limited performance in cases of varying illumination and complex terrain conditions. In this paper, we put forward a more advanced deep learning technique tailored for the purpose of crater and boulder detection to handle a diversity of shapes, sizes, and lighting conditions more robustly and accurately. This gives a holistic solution, adding data augmentation and learning transfer with universal applications under different conditions, unlike the current models, which might be applicable only to specific shapes or sizes of craters.

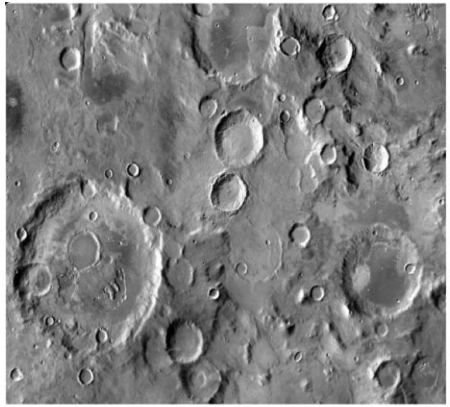
By applying state-of-the-art AI/ML models with a Deep CNN architecture, our solution is capable of automatically and accurately detecting craters and boulders, hence reducing the need for manual annotation, therefore accelerating the analysis process. Deep training on different datasets efficiently identifies craters and boulders with different characteristics. Furthermore, polygonal shapefiles and geospatial tools facilitate accurate mapping that enables geospatial analysis, and it is because of this that the output in the above magnitude becomes quite actionable to researchers.

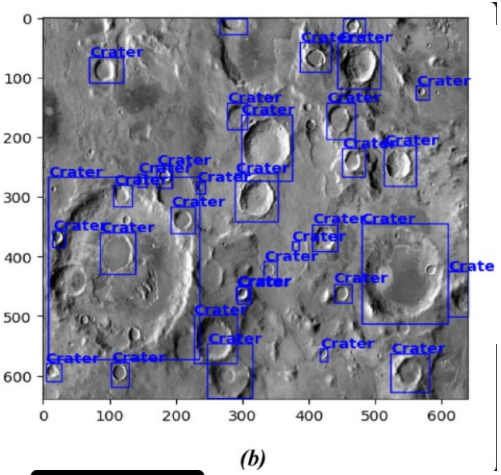
Our solution stands out in its sophisticated ability to automatically detect, measure and classify craters as well as boulders on OHRC images. It's able to perform well with several different shapes and sizes - almost no matter their illumination conditions, giving accurate selenographic positioning accuracy as well. This detailed, globally consistent and computationally compatible data set will improve the scientific good of lunar geologic maps for serial exploration missions by contributing through more accurate and relevant datasets to what is known about the nature of surface materials on Mare Tranquillitatis.

Proposed architecture/user diagram

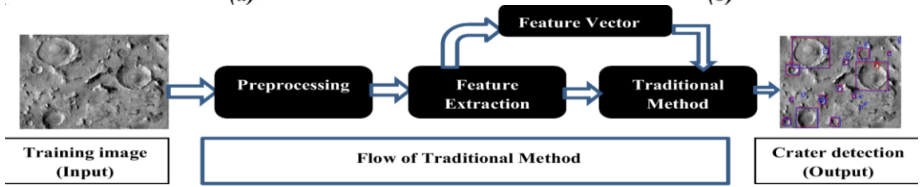


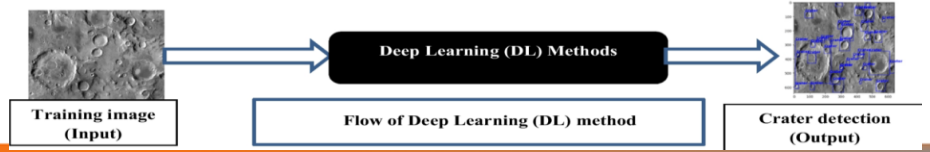
List of features offered by the solution

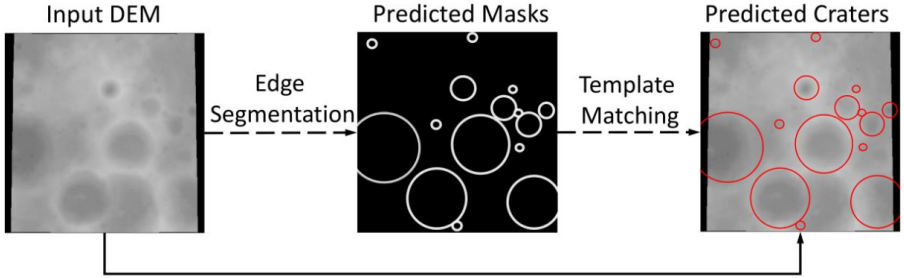


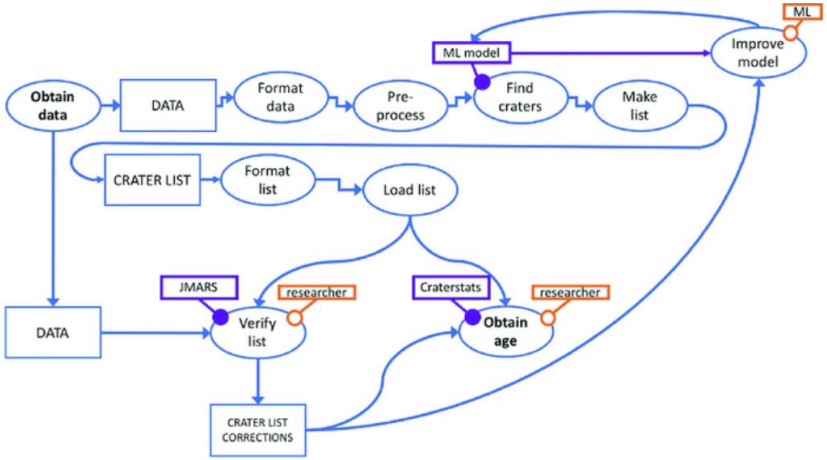


(a) (b)

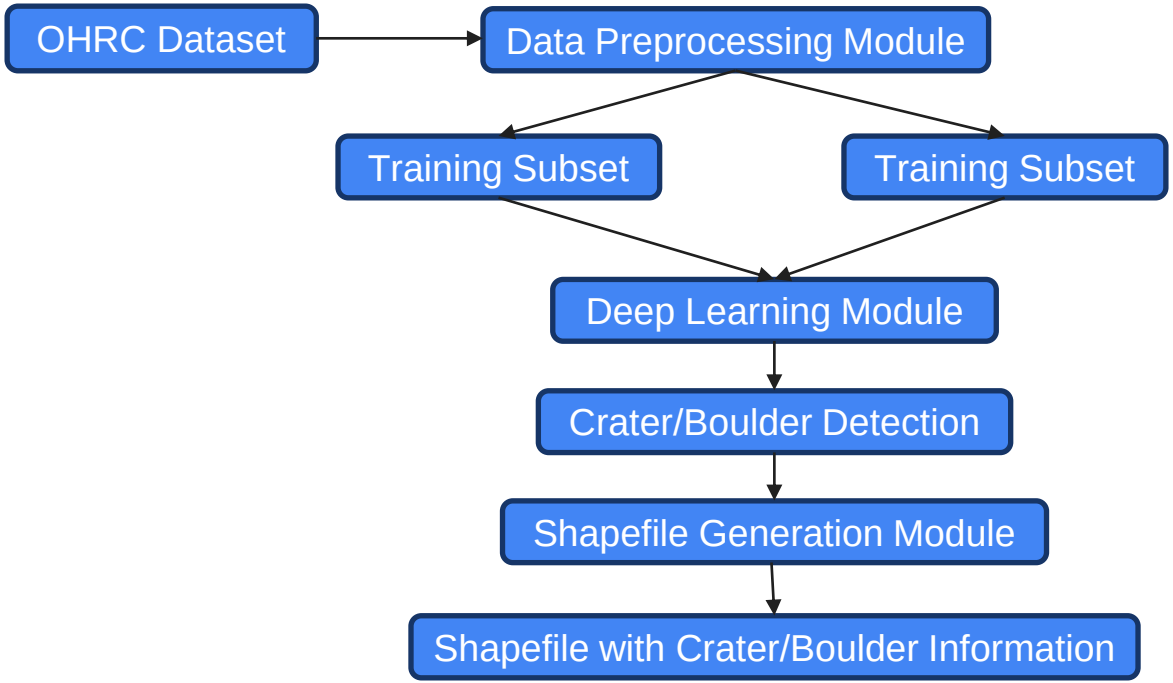








Process flow diagram or Use-case diagram

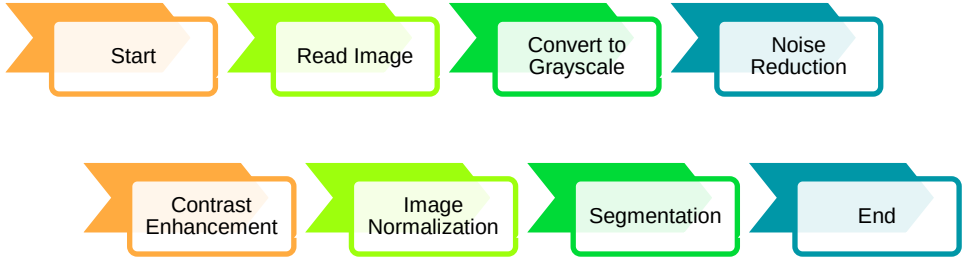


Data Preprocessing Module

```python
import cv2
import numpy as np

def preprocess_image(image_path):
    image = cv2.imread(image_path, cv2.IMREAD_GRAYSCALE)
    # Noise reduction
    image = cv2.GaussianBlur(image, (5, 5), 0)
    # Contrast enhancement
    image = cv2.equalizeHist(image)
    return image

preprocessed_image = preprocess_image('path_to_image.png')
```



Training and Testing Subset

```python
from sklearn.model_selection import train_test_split

# Assuming images and labels are already defined
X_train, X_test, y_train, y_test = train_test_split(images, labels, test_size=0.2, random_state=42)
```

Deep Learning Module

```
import tensorflow as tf
from tensorflow.keras import layers, models

def create_model():
    model = models.Sequential()
    model.add(layers.Conv2D(32, (3, 3), activation='relu', input_shape=(128, 128, 1)))
    model.add(layers.MaxPooling2D((2, 2)))
    model.add(layers.Conv2D(64, (3, 3), activation='relu'))
    model.add(layers.MaxPooling2D((2, 2)))
    model.add(layers.Conv2D(64, (3, 3), activation='relu'))

    model.add(layers.Flatten())
    model.add(layers.Dense(64, activation='relu'))
    model.add(layers.Dense(1, activation='sigmoid'))

    model.compile(optimizer='adam', loss='binary_crossentropy', metrics=['accuracy'])
    return model

model = create_model()
model.fit(X_train, y_train, epochs=10, validation_data=(X_test, y_test))
```

Crater/Boulder Detection

```python
def detect_craters(image, model):
    predictions = model.predict(image)
    # Assuming the model output is processed to find craters/boulders
    detected_features = process_predictions(predictions)
    return detected_features

detected_craters = detect_craters(preprocessed_image, model)
```

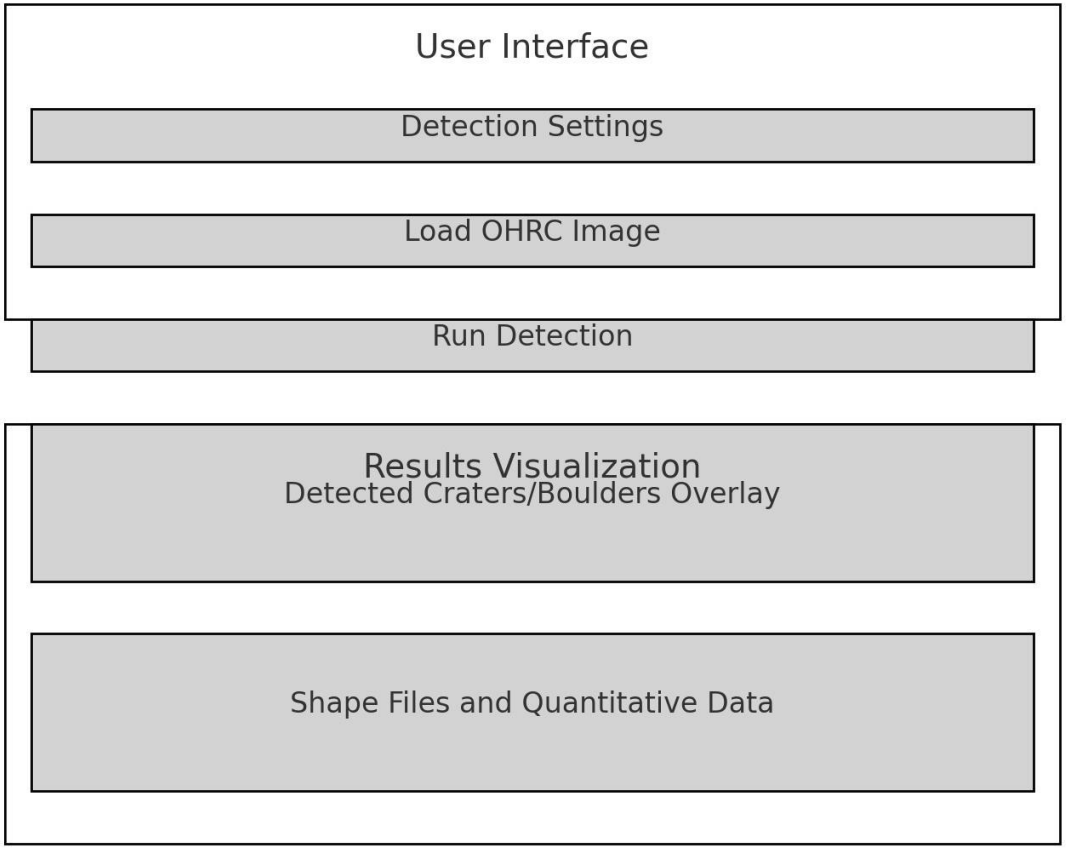
Shapefile Generation Module

```python
import shapefile

def generate_shapefile(detected_features, output_path):
    with shapefile.Writer(output_path) as shp:
        shp.field('ID', 'N')
        shp.field('Type', 'C')

        for i, feature in enumerate(detected_features):
            shp.point(feature['lon'], feature['lat'])
            shp.record(i, feature['type'])

generate_shapefile(detected_craters, 'output_shapefile.shp')
```


User Interface
Detection Settings
Load OHRC Image
Run Detection
Results Visualization
Detected Craters/Boulders Overlay
Shape Files and Quantitative Data

Solution Brief (Overall)

The automatic detection of craters and boulders in Orbiter High Resolution Camera (OHRC) images is crucial for planetary exploration and mapping

AI/ML techniques can significantly enhance the analysis of these high-resolution images.

Steps in the Solution

1. **Data Collection and Preprocessing**
    - **Data Collection:** Gather diverse OHRC images.
    - **Annotation:** Manually label a subset for ground truth.
    - **Data Augmentation:** Increase dataset diversity with techniques like rotation, scaling, and flipping.
2. **Model Selection**
    - **Convolutional Neural Networks (CNNs):** Use CNNs like U-Net, Mask R-CNN, or YOLO for feature extraction and object detection.
    - **Transfer Learning:** Fine-tune pre-trained models (e.g., from ImageNet) on OHRC images to reduce training time and improve performance.
3. **Feature Extraction**
    - **Multi-scale Analysis:** Capture both large craters and small boulders using multi-scale CNN architectures or image pyramids.
    - **Texture and Shape Analysis:** Utilize edge detection and Hough Transform for identifying circular shapes of craters.

4. **Model Training**
   - **Supervised Learning:** Train the model using annotated images, employing loss functions like Intersection over Union (IoU) for accurate localization.
   - **Regularization Techniques:** Prevent overfitting with dropout, batch normalization, and data augmentation.
5. **Model Evaluation**
   - **Metrics:** Use precision, recall, F1-score, and IoU to evaluate model performance.
   - **Cross-Validation:** Apply k-fold cross-validation to ensure generalization to unseen data.
6. **Post-processing**
   - **Noise Reduction:** Refine detections and reduce false positives with morphological operations.
   - **Clustering:** Use algorithms like DBSCAN to group detected features meaningfully.
7. **Deployment**
   - **Scalable Infrastructure:** Deploy the model on scalable cloud infrastructure using frameworks like TensorFlow Serving or PyTorch Serve.
   - **User Interface:** Develop an interface for scientists to upload images, run detections, and visualize results, including features for manual validation.
8. **Continuous Improvement**
   - **Feedback Loop:** Implement a mechanism for users to annotate and correct detections, feeding this data back for continuous improvement.
   - **Periodic Retraining:** Regularly retrain the model with new data and annotations to keep it updated.


BHARATIYA ANTARIKSH HACKATHON 2024
Innovation partner H2S
इसरो isro
THANK YOU