A Robust Deep Learning Model for Terrain Slope Estimation

Abdulaziz Alorf
Department of Electrical Engineering, College of Engineering,
Qassim University, Buraydah 52571, Saudi Arabia

***Abstract*—Interest in autonomous robots has grown significantly in recent years, motivated by the many advances in computational power and artificial intelligence. Space probes landing on extra-terrestrial celestial bodies, as well as vertical take-off and landing on unknown terrains, are two examples of high levels of autonomy being pursued. These robots must be endowed with the capability to evaluate the suitability of a given portion of terrain to perform the final touchdown. In these scenarios, the slope of the terrain where a lander is about to touch the ground is crucial for a safe landing. The capability to measure the slope of the terrain underneath the vehicle is essential to perform missions where landing on unknown terrain is desired. This work attempts to develop algorithms to assess the slope of the terrain below a vehicle using monocular images in the visible spectrum. A lander takes these images with a camera pointing in the landing direction at the final descent before the touchdown. The algorithms are based on convolutional neural networks, which classify the perceived slope into discrete bins. To this end, three convolutional neural networks were trained using images taken from multiple types of surfaces, extracting features that indicate the existing inclination in the photographed surface. The metrics of the experiments show that it is feasible to identify the inclination of surfaces, along with their respective orientations. Our overall aim is that if a hazardous slope is detected, the vehicle can abort the landing and search for another, more appropriate site.**

***Keywords*—Terrain slope estimation; spacecrafts; robotics; artificial intelligence; machine learning techniques; deep neural network; computer vision**

I. Introduction

In recent decades, interest in autonomous robots has significantly risen. Through the use of autonomous systems, we refer to systems that are conceived with capabilities to make certain types of decisions during the execution of their missions, minimizing the need for human-operator interventions. Their capability to accomplish missions that might be hazardous for humans (e.g., exploring celestial bodies or disaster monitoring), as well as to automate everyday tasks (e.g., driving cars or moving objects within a warehouse), make them highly valuable. They are currently the object of multiple research efforts addressing a wide range of challenges.

Space probes navigating in the proximity and landing on extraterrestrial celestial bodies [1], [2], [3] or indoor [4] and outdoor [5] navigating drones are among the most common applications of autonomous systems. For a robot intended to land on an asteroid (or other unexplored celestial bodies), the details of the terrain and local slope of its surface are very likely unknown, unless the celestial body had been previously studied thoroughly (e.g., the Moon or Mars) [6]. Moreover, as their missions might take place at large distances from Earth, real-time communication with ground stations might not be feasible, as the signals would require on the order of a number of minutes to traverse the path from the probe to Earth. Therefore, capabilities for autonomous navigation and landing are highly required. Likewise, vertical take-off and landing (VTOL) aircraft exploring unknown terrains also constitute highly demanded applications of these systems [7], [8]. In the aforementioned scenarios, landing on surfaces that are not well-known or mapped would require the lander to make decisions on how to approach and where to touch ground, thus minimizing the intervention of human operators [6], [9]. Fig. 1 shows an area of the Martian surface, with its many geographical features, including flat and inclined areas. A completely autonomous robot approaching the surface for landing should be able to determine whether it is going to touch down on a flat or inclined area.

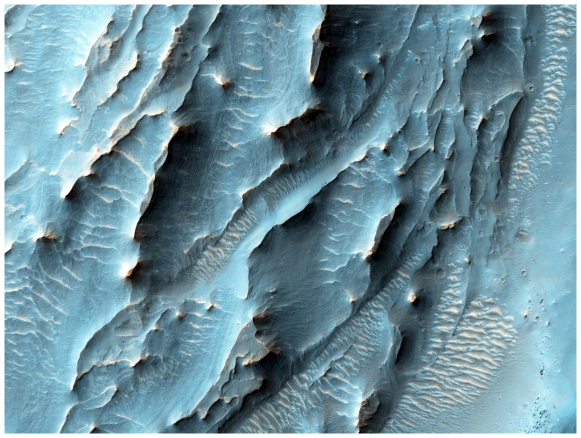


Fig. 1. Photograph of a piece of the martian surface. Credit: NASA/JPL-Caltech/Univ. of Arizona.

When a space probe or VTOL aircraft lands, the final descent to the surface intends to follow position and attitude trajectories that ensure a smooth touchdown, with the entire landing gear leaning simultaneously on the ground. In general, these trajectories aim to touch ground at areas where a vector normal to the local terrain is somewhat aligned with the negative local gravity vector [6], [10]; however, there have been some efforts to study the feasibility of take-off and landing from sloped terrain [11]. Fig. 2 illustrates this concept.

In Fig. 2, there are two landers, one denoted by A (on the left hand-side of the figure), and the second denoted by B (at the center of the figure). The arrows labeled as DM show the

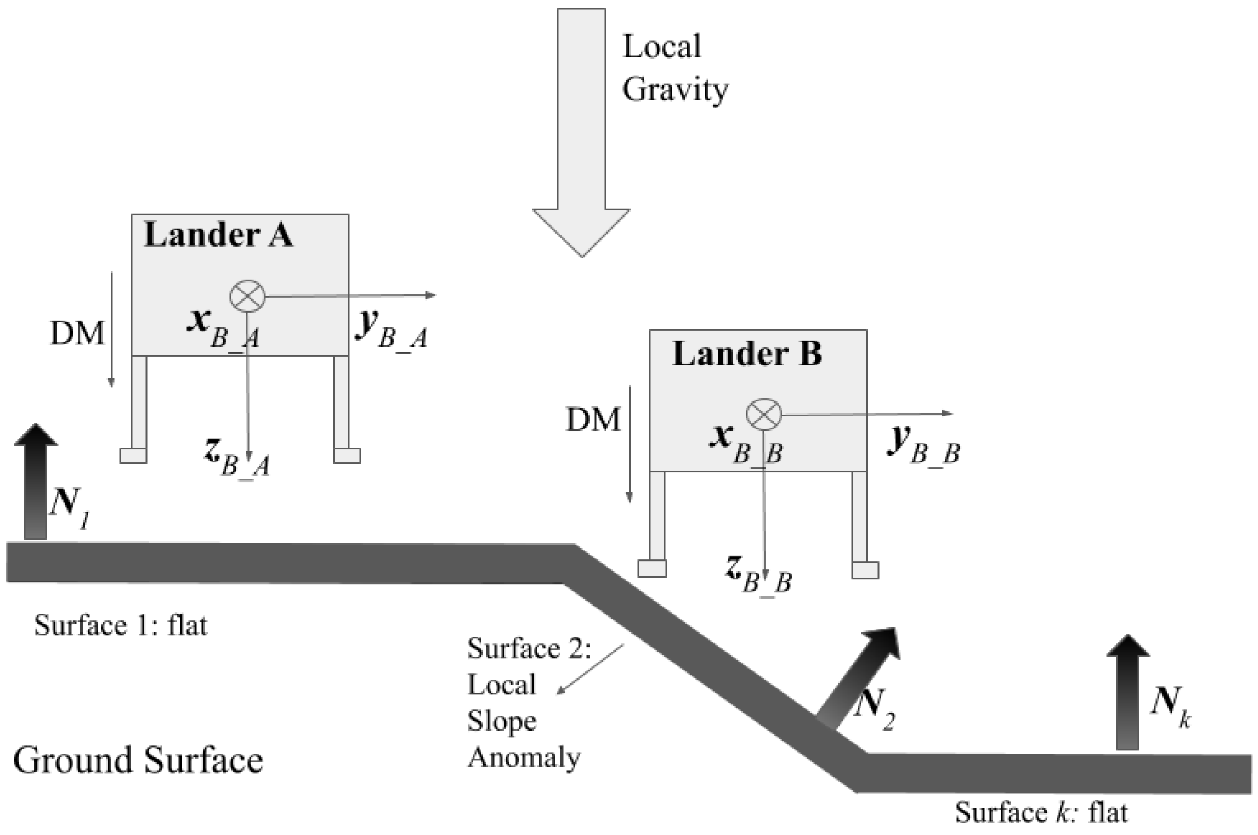


Fig. 2. Schematic view of robots landing on flat and inclined terrain.

direction of motion of both landers, and the local gravity vector is also depicted. The two dextral coordinate systems, denoted as B, fixed to the bodies of the landers A and B, respectively, can also be observed. The unit vectors $\boldsymbol{z}_B$ always point along the legs of the body, while the unit vectors $\boldsymbol{y}_B$ point towards the right side of it, as can be seen in Fig. 2. Correspondingly, $\boldsymbol{x}_B = \boldsymbol{y}_B \times \boldsymbol{z}_B$.

A hypothetical piece of terrain where the probes would land is also displayed in Fig. 2. Assume that the piece of terrain consists of multiple piecewise planar surfaces, k, that are continuously concatenated, and each of which has a different slope represented by a local normal unit vector $\boldsymbol{N}_k$ associated with it. The depicted scene shows Lander A about to touch ground on an area with $\boldsymbol{z}_B = -\boldsymbol{N}_1$. In contrast, Lander B is about to touch ground on a highly inclined region with respect to $\boldsymbol{z}_B$. This may cause the lander to touch ground on its left leg. If the lander ignores the inclination of the surface upon touching ground, it might constitute a hazard for the landing moment. The robot would touch ground with only its left leg, perhaps overloading it; alternatively, it could turn over, roll down, or even slip down, leading to an undesired termination of the mission. For sake of clarity, $\boldsymbol{z}_{B\_A}$ and $\boldsymbol{z}_{B\_B}$ will hereafter be indistinctly referred to as $\boldsymbol{z}_B$.

In order to avoid these hazards, these robots should be endowed with the capability to sense that the surface underneath is highly inclined, with respect to its attitude (as with Surface 2); and, hence, that it is necessary to navigate to a more suitable location for touching ground (as with Surface 1). Understanding information on the terrain and the slope of the surface underneath constitutes an essential task in any of the following cases: (I) A team at the control center makes the decision on where and how to land, and imparts the commands to the robot; or (II) the robot decides by itself where to land and how to accomplish it.

Current and past missions have used distance-based sensors, such as a radar or lidar [12], [13], [14], in order to evaluate the slopes of the terrain underneath. In both cases, electromagnetic waves are sent towards an object or surface, with respect to which the measuring distance is determined.The time at which the reflection of these waves reaches the sensor is measured, which enables the computation of the distances to multiple points. An advantage of these sensors is the accuracy with which they can measure the terrain; however, they are usually expensive and power-consuming. Motivated by the aforementioned scenarios, we explore the feasibility of using individual images from a monocular camera (in the visible spectrum) to classify the relative inclination between the focal plane of the camera (associated with the vector $\boldsymbol{z}_B$ in Fig. 2) and the piece of terrain photographed by the camera (associated with the vector $\boldsymbol{N}$ in Fig. 2). This contribution would allow a lander to use monocular images to evaluate the inclination of the terrain underneath and assess whether or not it is a good site to touch ground. This work proposes an alternative methodology to evaluate the slope of the terrain, based on the processing of monocular images with artificial neural networks. The aim of this concept is not to substitute the usage of radars and lidars, but to assess another operational principle that could be used on its own or fused with information retrieved from the aforementioned sensors. To the best of our knowledge, we could not find approaches similar to the one presented herein, which is why we considered it interesting to evaluate the feasibility of the approaches elaborated in the following.

A. Related Work

Computer vision applications for space vehicle navigation have been pursued intensively in recent decades [15]. In this respect, photo-cameras also constitute sensors that can be used for relative attitude determination and position. When two or more spacecraft are flying in proximity or performing docking maneuvers, images in the visible spectrum can provide highly relevant information to determine the relative position and attitude between them [9], [16], [17]. Images or videos recorded by the cameras are processed by algorithms that identify feature points (which might be pre-defined or not) and track them along the sequence of images. The position of these feature points in the images provides an indication about the relative position and orientation between the two spacecraft during the maneuvers. In these types of applications, the information retrieved from images is usually fused with information provided by other sensors, such as gyros, range finders, and star trackers.

In the context of aerial vehicles, the use of computer vision techniques has also been intensively explored, especially for landing and collision avoidance purposes. In [18] and [19], the usage of optical flow measurements for deriving control laws for landing VTOL unmanned aerial vehicles on moving platforms was investigated. In study [20], optical flow measurements were exploited for the determination of landing control laws of UAVs in cluttered environments. In study [21] and [22], optical flow was also used, but for collision avoidance purposes. In study [23], neural networks were implemented to process video signals for indoor navigation assistance in maneuver planning. In study [24], a thorough survey of the vision-based techniques used for UAV navigation was presented.

There have also been efforts to determine terrain slopes from multiple overlapping aerial or satellite imagery [25], [26], [27], [28], [29], [30], [31]; however, these articles were not specifically intended for implementations in autonomous systems that make decisions in real-time from individual images, as they need to combine multiple overlapping images to determine the slope of the photographed terrain.

Extracting 3D features from 2D scene projections (images) has been considered a central challenge in computer vision. It has been extensively tackled in a diversity of contexts, and through multiple approaches. Among them, using texture cues to understand the shapes of 3D geometries has been highly pursued [32], [33], [34]. In research [35], the recovery of 3D shapes from the observed distortions in the density of the textures was analyzed, and the corresponding equation for determining the shapes of planar and curved surfaces was derived. In study [36], [37], and [38], affine transforms were proposed to model the relationship between the texture distortion and direction of the points in the image. Using these transforms, the orientation and shape parameters were estimated for multiple directions in the image. In study [39], the authors adopted 3D morphable models that are fitted to pixel intensity, edges, and specular highlights, thus maximizing the posterior probability of the parameters upon the input image.

The estimation of depth (the coordinate along the line of sight) from monocular images also constitutes another highly pursued technical challenge, as local features do not represent enough information to estimate the depth of arbitrary points in images. Usually, depth is perceived as a result of two or more associated vision sensors (stereo-vision). In study [40], a Markov Random Field was used to learn depth cues from monocular images, which were used to reinforce a stereo vision system. In study [41], depth maps for still scenes were estimated, based on depth cues captured by supervised learning algorithms. In study [42], using the Lucas–Kanade method, the authors aimed to estimate the depths in scenes captured by a moving camera. In research [43], the authors exploited de-focus (blur) and textures to estimate depth maps. In study [44], a CNN was utilized to estimate the relative and absolute depth maps, which were optimally combined. In study [45], a ranking approach was used for relative depth estimation.

Terrain characterization can be considered a specific application of 3D shape estimation. In robotics navigation, terrain characterization is essential for the landing and ground traversal of robots performing on unknown terrains. Computer vision applications have also been extensively used for this purpose. In study [46], the authors proposed a system for terrain recovery which could be used for autonomous rotorcraft. The system aims to recover the material properties and geometry of the local terrain, using stereo cameras, a global navigation satellite system (GNSS), and inertial measurement units (IMU). In study [47], principal component analysis was used to estimate the normal vectors of surfaces, in order to determine traversable and non-traversable areas from depth images. This method was implemented in a rover-like robot, equipped with cameras and depth sensors, which allowed for the measurement of three-dimensional point clouds. In researches [48] and [49], algorithms based on convolutional neural networks (CNN) were presented for the classification of different terrain types and surface features, from both orbital and ground images. These outcomes were then used to determine terrain traversability for rovers. Terrain classification has been also pursued based on proprioceptive signals [50].

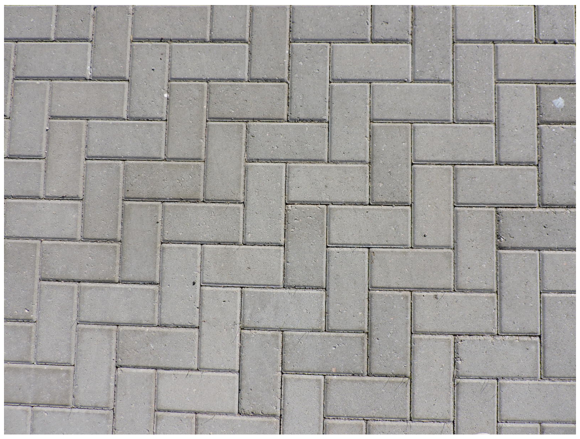
This work attempts to use convolutional networks to extract features from monocular images, thus allowing for estimation of the slope of the piece of terrain just below a landing vehicle. We presumed that a properly trained neural network can find, in monocular images, indicators of the slopes and orientation of the terrain below. These cues would mainly be in the variations of the visual textures (densities and sizes) across an image. To further define the problem tackled in this article, several more concepts are introduced.

B. Incidence Angle, Far and Near Sides

In many scenarios where the human eye is looking at a given surface, the brain can understand whether the surface is normal or inclined with respect to the line-of-sight (LOS)[1].

[1]By line-of-sight, we refer to the line joining our eyes (or the camera) with the aimed object

This is an ability that we learn when our vision system commences its development, which improves as we are exposed to more types of surfaces with different inclinations. We do not have the capability to accurately measure the relative angle between the LOS and the plane of the surface we are looking at. However, we can distinguish qualitatively high relative inclinations from low or null inclinations. For instance, consider Fig. 3a and Fig. 3b, which show images of a street surface paved with stones, taken at different angles. i denotes the incident angle, which is defined as the angle between the LOS and the vector $\boldsymbol{N}$ normal to a given surface.


(a) Incident angle $i \simeq 0$.

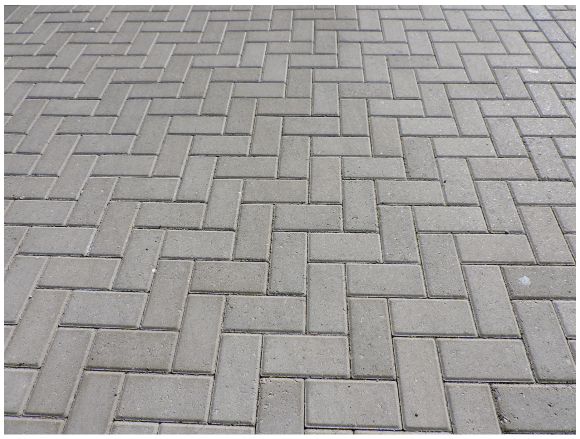
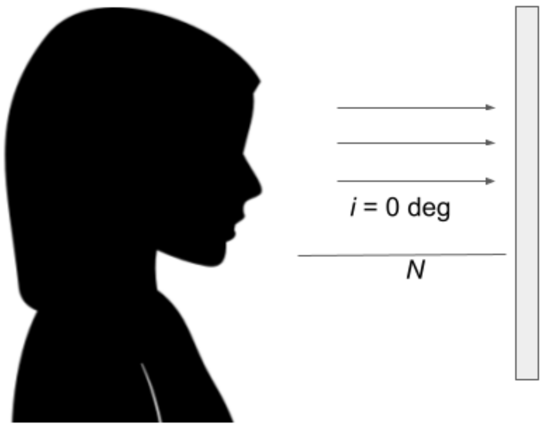

(b) Incident angle > 0.

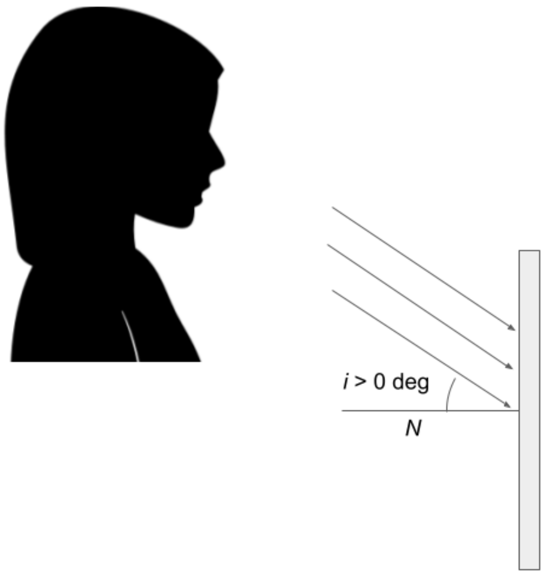
Fig. 3. Surfaces perceived as looking with different incident angles.

Fig. 4a and Fig. 4b show a person looking at a surface from $i = 0$ and $i > 0$ deg, respectively. Generally speaking, a person could determine, only from an image, whether $i > 0$ or not. The reader could probably determine that the surface shown in Fig. 3b has a higher incident angle, with respect to the surface of the street, than that of Fig. 3a. Clearly, the texture of the surface we are looking at helps us to distinguish the angle of incidence. A purely plain surface with an incident angle would be impossible to distinguish from the same surface with a null incident angle.

Furthermore, it can also be seen, from Fig. 3b, that the upper part of the image was at a larger (far side) distance



(a) Null incident angle, $i = 0$.



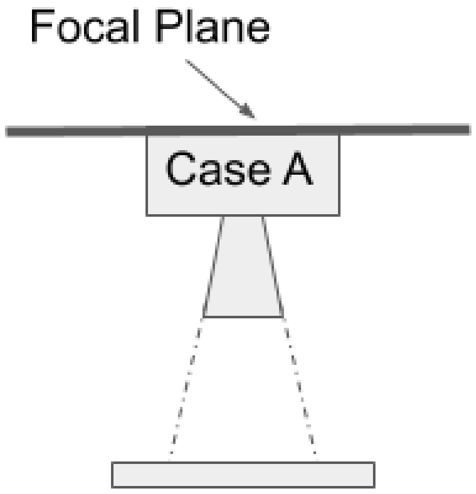
(b) Incident angle > 0.

Fig. 4. Person looking at a surface from different incident angles.

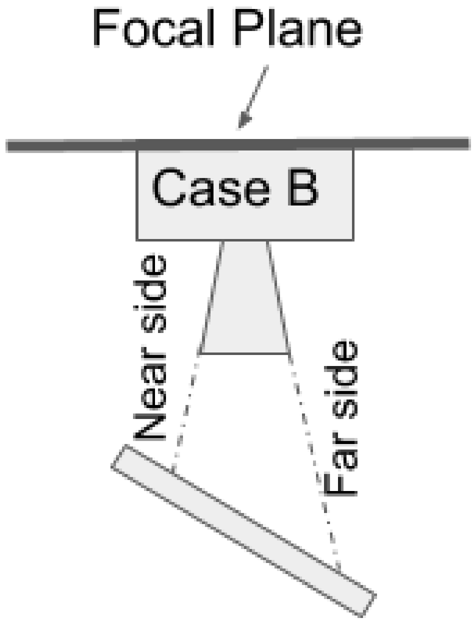
from the focal plane of the camera than the lower part (near side) of it. The *far side* and *near side* of an image denote the sides of it that are farther from and closer to the focal plane, respectively. This is illustrated in Fig. 5a and Fig. 5b. Fig. 5a illustrates the case where the angle of incidence of the camera with respect to the surface is null, which would correspond to the Lander A of Fig. 2 if it had a camera pointing along its $\boldsymbol{z}_B$ axis. Fig. 5b represents a scenario where the angle of incidence is considerably greater than zero, which would correspond to what would be seen by a camera pointing along $\boldsymbol{z}_B$ in Lander B. In the latter case, the far and near sides are indicated in the image.

Moreover, for images characterized by $i > 0$, we define a roll angle, r, as the angle measured clockwise between a vector from the center of the image pointing towards the far side of it and a vector from the center of the image pointing towards the upper edge of it.

With the definitions stated above, the goal of this article is to report two experiments, aimed at deriving algorithms that



(a) Null incident angle, $i = 0$.

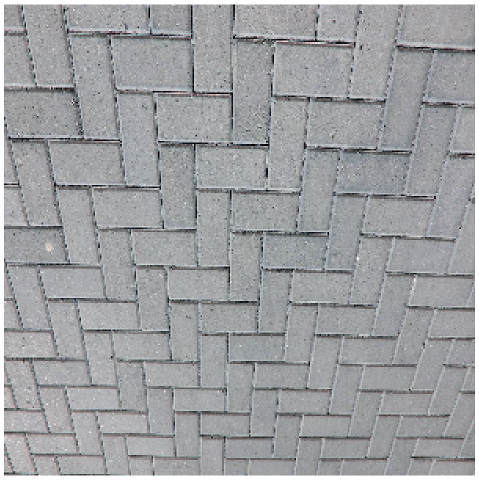


(b) Incident angle, $i > 0$.

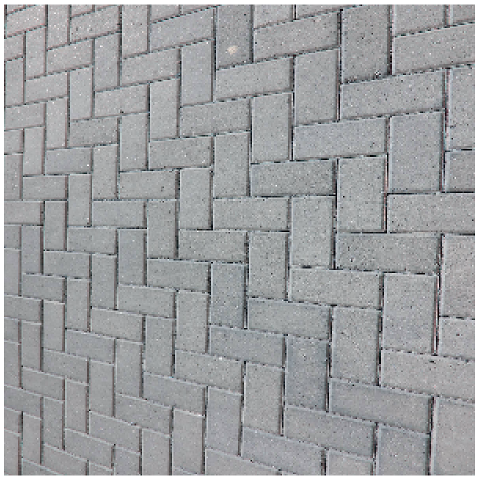
Fig. 5. Difference in geometries for images taken with $i = 0$ (case A) and with $i > 0$ (case B).

solve the following problem: *With a single monocular image of a surface in the visible spectrum, classify the angle between the normal vector at the photographed surface and the optical axis of the camera (i.e., z_B) using the following categories:*

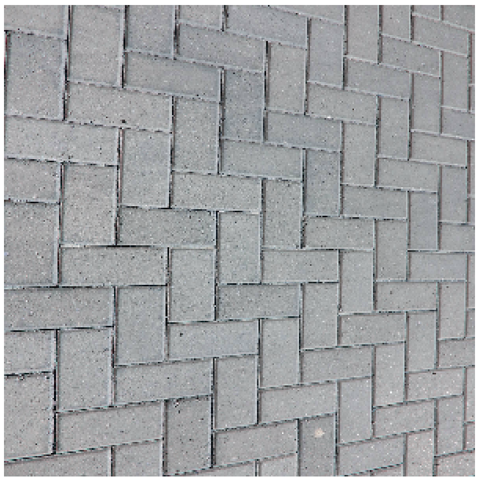
- *Normal* (no inclination), r undefined, denoted as type N, as seen in Fig. 3a;
- *Upward* inclination, $r = 0$ deg, denoted as type UI, as seen in Fig. 3b;
- *Downward* inclination, $r = 180$ deg, denoted as type DI, as seen in Fig. 6a;
- *Leftward* inclination, $r = 270$ deg, denoted as type LI, as seen in Fig. 6b;
- *Rightward* inclination, $r = 90$ deg, denoted as type RI, as seen in Fig. 6c.



(a) Surface with incident angle $i > 0$ and $r = 180$ deg.



(b) Surface with incident angle $i > 0$ and $r = 270$ deg.



(c) Surface with incident angle $i > 0$ and $r = 90$ deg.

Fig. 6. Different angles r for images with $i > 0$ deg.

A few observations follow. Tackling the problem as a classification problem constitutes a coarser measurement and is fairly easier than determining the angles precisely as continuous real numbers. However, it is an essential first step in the major goal of an ongoing project, aimed at precisely estimating these angles as real continuous variables. Indeed, one of the described experiments classifies the images considering three ranges of values for the incidence angle: 0 deg, 20 deg, and 40 deg. The two aforementioned experiments are elaborated in the following sections.

In order to derive a solution to the problem stated above, we trained convolutional neural networks using images of diverse types of surfaces. These surfaces represent ground terrains where a lander might perform a touchdown. The appearance of a landing surface can be very diverse. Therefore, we intended to obtain an algorithm that can work well with multiple textures, including some that can be found in the Earth's ground types and others that are not necessarily observed as ground surfaces. Including images with multiple textures allows the CNNs to learn features that can provide information about the angles i and r, even when the angle is sensed across images having different textures. Section II-C displays samples of the types of surfaces used in this work.

Artificial neural networks (ANNs) have been used extensively for classification problems [51]. They represent algorithms that can be very efficient for solving high-dimensional classification problems. Once they have been trained, they can process given inputs and return the probabilities that these inputs to belong to any of the classes for which they have been trained. Training refers to the process that optimizes the internal coefficients of the operator to the specific classification process intended [52], [53].

In the field of image processing, CNNs constitute a powerful tool for image classification and interpretation [54]. CNNs are special types of NN that are well-suited to dealing with images. The images are introduced as tensors, where each entry of the tensor dictates the color intensity of its corresponding pixel. Upon every input image, the CNN performs a sequence of mathematical operations on the numerical values assigned to each pixel, and computes the probability of the whole image (or a part/parts of it) corresponding to a given pre-defined class.

CNNs are mainly composed of convolutional layers and, possibly, other types of intermediate layers. A convolutional layer is a portion of the algorithm where convolutional filters are applied to the input of that layer. By means of these convolutional filters, CNNs perform convolutional operations on these images, extracting features that provide indications about which of the pre-defined classes best characterize the image analyzed [55].

Typical applications of CNNs include object recognition within images [56], determining the position of sought objects within images [57], [58], and identifying pathologies such as CoViD-19 in pulmonary radiographies [59], [60], [61]. CNNs can have multiple architectures. The architecture of a CNN refers to the sequence in which the multiple operations are applied to the input. In a given classification problem, different architectures can produce different results, with the same inputs. In this work, we intend to exploit these classification tools to determine which of the aforementioned classes would best characterize an image, thereby associating each image with the most suitable ranges for i and r.

The approach proposed herein should enable a lander robot to interpret how appropriate for landing, in terms of inclination, the piece of terrain underneath it is. To date, CNNs have not been used to analyze angles between the focal plane and the photographed surface. The contribution of this work consists of a methodology to create algorithms that can classify the incidence angle of an image into certain pre-defined categories. These algorithms could become an essential tool for space probes or autonomous VTOL aircraft, in order to determine whether a piece of terrain has a slope that makes it appropriate landing site.

The remainder of the paper is structured as follows: Section II describes the image collection and preparation processes, as well as the architecture of the convolutional neural network implemented for the classifier. Section III elaborates on the obtained results and discusses potential directions for improvement. Finally, Section IV presents our main highlights and observations in this work.

II. Methodology

This work reports two experiments in which, using CNNs, we address two versions of the problem stated in Section I. These experiments are referred to as Experiment I and Experiment II, and are described in the following. The proposed pipeline, which represents the procedures followed by the experiments, is shown in Fig. 7. The figure shows the process of collecting images, augmenting the data set, and training and testing the respective convolutional neural networks.

A. Experiment I

This experiment aimed to derive algorithms to solve the following problem: *With a single image of a surface, taken by a single camera in the visible spectrum, classify the angle between the normal vector at the photographed surface and the optical axis of the camera (i.e., $\boldsymbol{z}_B$) into the following categories*:

- *Normal* (no inclination), r undefined, denoted as type N, as seen in Fig. 3a;
- *Upward* inclination, $r = 0$ deg, denoted as type UI, as seen in Fig. 3b;
- *Downward* inclination, $r = 180$ deg, denoted as type DI, as seen in Fig. 6a;
- *Leftward* inclination, $r = 270$ deg, denoted as type LI, as seen in Fig. 6b;
- *Rightward* inclination, $r = 90$ deg, denoted as type RI, as seen in Fig. 6c.

As previously stated, these problems were tackled using CNNs. At present, there exist several CNN architectures that are renowned for having very good performance in certain types of image classification problems. To address this experiment, we implemented two different architectures and compared the exhibited performance. These architectures are described in Section II-A1 and Section II-A2.

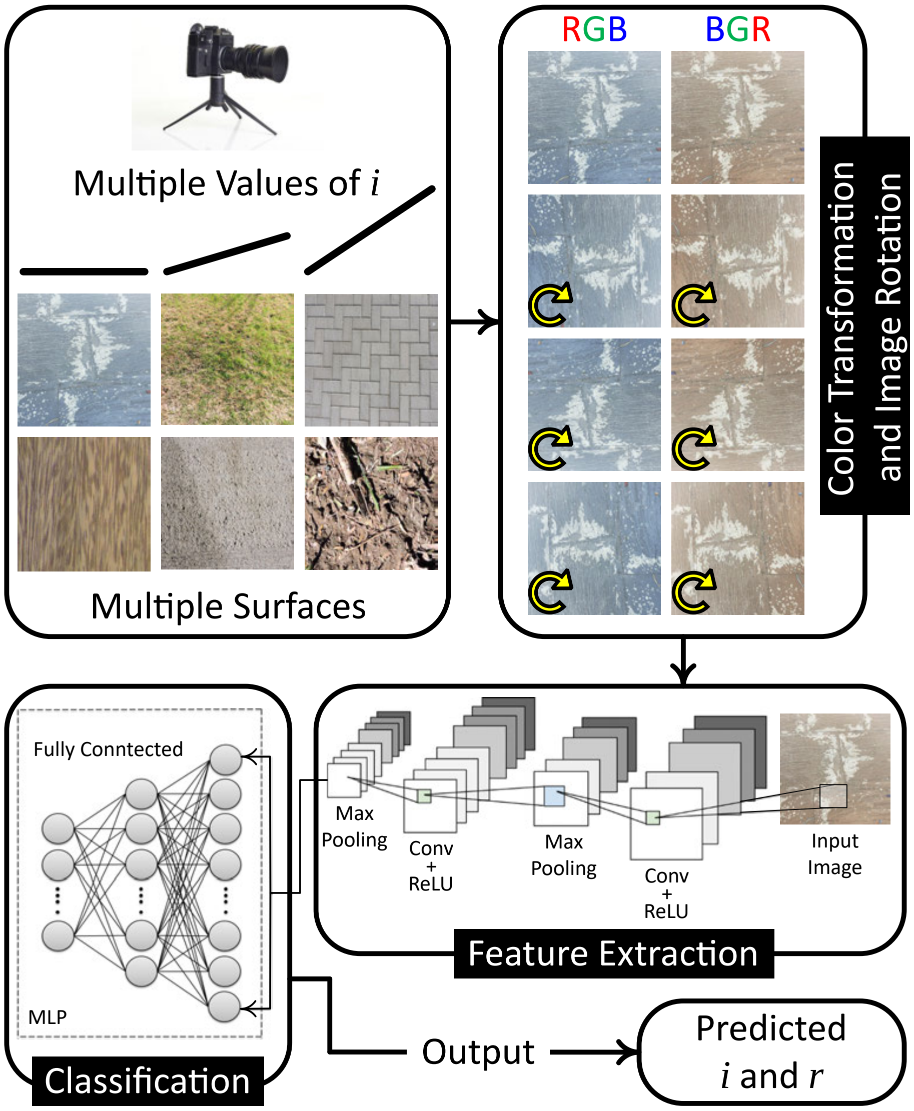


Fig. 7. Schematic representation of the pipeline which represents the procedures followed by the experiments.

1) Convolutional neural network based on the VGG16 architecture: First, we implemented a convolutional neural network based on the VGG16 architecture. This architecture was proposed by the Visual Geometry Group at the University of Oxford, winning the ILSVR (Imagenet) competition in 2014 [56]. Since then, it has become widely used, thanks to its excellent performance. It was described in the seminal work by Simonyan and Zisserman [62] and, since then, has been implemented in a variety of applications [63], [64], [65]. A schematic view of the architecture of this CNN is presented in Fig. 8.

In its original implementation, the input of the first convolutional layer had a fixed size of 224×224 and was of RGB type. The image was processed through an array of 13 successive convolutional layers with max pooling layers in between every two or three convolutional layers, as displayed in Fig. 8. Convolutional filters of 3×3 were used in every convolutional layer, with stride of 1 pixel and padding of 1 pixel. Spatial pooling was performed in a total of five max-pooling layers. The max-pooling was performed through windows of 2×2 pixels and a stride of 2 pixels. After the convolutional layers, three dense layers with 4096 nodes each were used, followed by the final softmax layer. The hidden layers were built using rectified linear unit (ReLU) activation functions.

In this work, the implementation of the VGG16 network was accomplished using the Keras API [66]. Keras is a programming interface for Tensorflow, which is a Google open-source library for machine learning applications [67]. Keras has a built-in implementation of the VGG16 network that allows for image sizes different from the original configuration of 224×224 pixels, enabling us to define a model using an image size of 300×300. Keras also enables the user to remove the three fully connected layers at the end of the network that the original version of VGG16 had. In this work, these layers were removed, and the output of the convolutional layers was passed through a filter of the average pool type with a size of 2×2 pixels. Subsequently, a single fully connected layer of

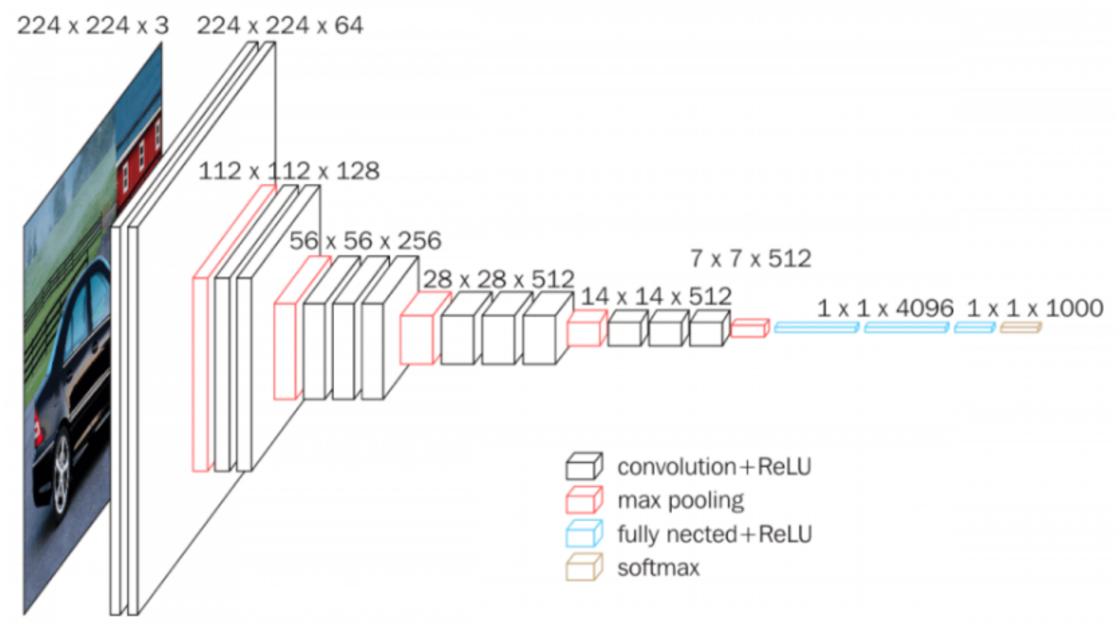


Fig. 8. Schematic diagram of the VGG16 CNN architecture. Credit: https://neurohive.io/en/popular-networks/vgg16/.

128 nodes was added with the ReLU activation function and, finally, a five-node output layer with softmax activation was used [68].

2) Convolutional neural network based on the xception architecture: Another architecture assessed in this work is Xception. Xception was developed at Google by F. Chollet. It is based on the concept of "inception modules," and is endowed with 36 convolutional layers for feature extraction [69]. This architecture seems very promising, as it outperformed other precedent networks with similar number of parameters on large data sets such as ImageNet [56] and another Google internal data set denoted by JFT.

Like VGG16, the Xception architecture can be also implemented through the Keras framework [70]. In its default implementation, the size of the input images is 299 pixels $\times$ 299 pixels, with 3 channels. However, disabling the default top layer of this CNN allows for the use of images with other sizes. We opted to disable the default top layer, in order to keep the same input size of 300 pixels $\times$ 300 pixels, and replaced the disabled layer by another fully connected layer with 128 nodes.

In the resulting architecture, the image is processed through an array of 14 blocks of successive convolutional layers. Each block is composed differently, combining *depth-wise separable* convolutional layers with ReLU activation functions, and 3×3 max-pooling layers. As with the previous architecture (Section II-A1), the output of the convolutional layers was passed through a filter of the average pool type, with a size of 2×2 pixels and a fully connected layer of 128 nodes with the ReLU activation function. Likewise, the output layer had five nodes with softmax activation.

3) Training and testing sets: In this experiment, the whole data set consisted of 1500 images, which included 300 images of type UI, 300 images of type DI, 300 images of type LI, 300 images of type RI, and 300 images of type N. The training set was constructed including 250 images of each class, while the remaining images (50 from each class) constituted the testing set.

In order to generate the images of class N (null inclination), the mobile was held parallel to the surface, with an allowed error up to 3 deg (i.e., $i \leq 3$ deg). On the other hand, the images that were not intended to be of class N (i.e., with inclination) were taken with i within a range of 30–70 deg. With respect to r, for every class other than N, we allowed it to include values of r centered at their nominal values r_0 (0, 90, 180, or, 270 deg) and within intervals of $r \in [r_0 \pm 5\,\text{deg}]$.

4) Training process: For both CNN architectures, the weights that were optimized were those of the fully connected layers, while the weights of the convolutional layers were set as the default pre-trained values.

The optimization of the weights was achieved using the Adam optimizer [71], for which the learning rate lr was scheduled as $lr\,(k) = 0.001/\,(1 + d \cdot k)$, where k is the iteration number and $d = 0.00002857142$. The loss function $\mathcal{J}$ was categorical cross-entropy. Both models were trained for 40 epochs with batches of 30 observations.

B. Experiment II

Experiment II was considered as a natural step forward from Experiment I. In this case, more refined categories for i were pursued. We intended to obtain an algorithm that could distinguish incidence angles in three classes: $i_0 = 0$ deg, $i_0 = 20$ deg, and $i_0 = 40$ deg. The sub-index 0 indicates the nominal value for each corresponding class. The real images were taken with some errors allowed (for i up to 3 deg) from their corresponding nominal values. In other words, the class with $i_0 = 0$ deg actually contained angles $i \leq 3$ deg. The class of $i_0 = 20$ deg implied 17 deg $\leq i \leq 23\,\mathrm{deg}$, while that of $i_0 = 40$ deg included angles in the range 37 deg $\leq i \leq 43\,\mathrm{deg}$. For each class with $i_0 > 0$ deg, the algorithm had to distinguish between $r_0 = 0$ deg, $r_0 = 90$ deg, $r_0 = 180$ deg, and $r_0 = 270$ deg. In total, there were nine classes, denoted by the following: 'N' ($i_0 = 0$ deg), '20U' ($i_0 = 20$ deg and $r_0 = 0$ deg), '20R' ($i_0 = 20$ deg and $r_0 = 90$ deg), '20D' ($i_0 = 20$ deg and $r_0 = 180$ deg), '20L' ($i_0 = 20$ deg and $r_0 = 270$ deg), '40U' ($i_0 = 40$ deg and $r_0 = 0$ deg), '40R' ($i_0 = 40$ deg and $r_0 = 90$ deg), '40D' ($i_0 = 40$ deg and $r_0 = 180$ deg), and '40L' ($i_0 = 40$ deg and $r_0 = 270$ deg).

In this experiment, the architectures described in Section II-A1 and Section II-A2 were initially attempted, but their results were not promising. Hence, a third architecture was pursued. This architecture was proposed in [68] and, due to its similarities to the VGGNet architecture [62], it is referred to as the compact version of VGGNet, named *SmallerVGGNet*. The following section provides a description of it.

In this experiment, rather than considering nine mutually exclusive classes, the problem was addressed as a multi-label classification process. Each image was assigned to two labels. One label indicated the incidence angle: either $i_0 = 0$ deg, $i_0 = 20$ deg, or $i_0 = 40$ deg. For the cases with $i_0 > 0$ deg, another label expressing the angle r_0 was associated with them: either $r_0 = 0$ deg, $r_0 = 90$ deg, $r_0 = 180$ deg, or $r_0 = 270$ deg. Thereby, by using the CNN to classify the images on i_0 and r_0, all of the cases of interest were addressed.

1) SmallerVGGNet architecture: A schematic representation of this architecture can be found in [68]. Following the input layer, it has a first convolutional layer with 32 kernels of size 3×3, activated by the Rectified Linear Unit (ReLU) function. It uses a padding of 1 and stride of 1. This layer is followed by a MaxPool layer of size 3×3 and a stride of 3×3. A dropout scheme with a rate of 25% is applied before the next convolutional layer. Then, there are two convolutional layers, each of which has 64 filters of size 3×3, and a ReLU activation function. These are followed by another MaxPool layer of size and stride 2×2. Another dropout layer with a rate of 25% was applied before the next convolutional layer. These layers are followed by another two convolutional layers, with 128 kernels each, where each kernel had a size of 3×3. These convolutional layers are activated with ReLU functions, and are concatenated to another MaxPool layer, of size and stride 2×2, and a dropout layer with rate 25%.

Following the aforementioned layers, there is a fully connected layer of 1024 nodes with ReLU activation, a dropout layer with a rate of 50%, and the final output layer activated with sigmoid functions. It is important to note that, for Experiment II, the problem was tackled as a multi-label classification one, categorizing both angles i and r independently.

2) Training and testing sets: In this experiment, the training set consisted of:

- 1252 images of type N;
- 293 images of type 20U;
- 282 images of type 20R;
- 289 images of type 20D;
- 292 images of type 20L;
- 289 images of type 40U;
- 301 images of type 40R;
- 284 images of type 40D; and
- 288 images of type 40L.

Meanwhile, the testing set contained:

- 220 images of type N;
- 45 images of type 20U;
- 56 images of type 20R;
- 49 images of type 20D;
- 46 images of type 20L;
- 55 images of type 40U;
- 43 images of type 40R;
- 60 images of type 40D; and
- 56 images of type 40L.

The number of images of type N might seem much higher than that for the other classes. This is due to the fact that, when the initial set of images was augmented by rotating them, all of the rotated images within the category N remained in the same category. All these images were used with the intention to provide images in category N with different roll angles.

3) Training process: The CNN was trained for initially 30 epochs, with batches of 32 images each. Optimization of the weights was achieved by using the built-in Adam optimizer [71], for which the learning rate lr was scheduled as $lr\,(k) = 0.001/\,(1 + d \cdot k)$, where k is the iteration number and $d = 0.00002857142$. After the first 30 epochs, the CNN was retrained for another 10 epochs, but with a learning rate given by $lr\,(k) = 0.0005/\,(1 + d \cdot k)$.

C. Images and Surfaces Considered

The images used in this work for training show multiple type of surfaces with different textures. They represent ground terrains where a lander might accomplish touchdown. The appearance of surfaces where a space probe might land can be very diverse. There might be areas with multiple protruding rocks, or areas that are mostly plain [72]. For VTOL aircrafts, the landing surface could be also very varied, including grass, concrete, and paving stones. We intended to obtain an algorithm that can work with multiple textures, including some that can be found on the Earth's ground and others that are not of ground-like types. Including images with multiple textures allowed the CNN to learn features that can provide indications about different inclinations, even across images of different textures. Samples of these surfaces are shown in Fig. 6c, and Fig. 9a–Fig. 9l.

The images were taken using a mobile telephone camera. The device was a Samsung® S6 Edge. The resolution of the camera was 16 megapixels. While taking the images, the mobile was held manually, at a distance between 30 and 100 cm from the surface. However, for missions, the distance

(a) Grass surface, $r = 270$ deg.

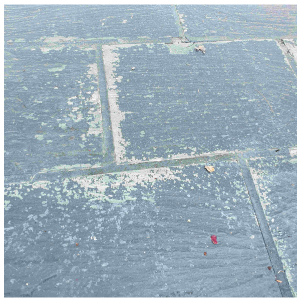
(b) Concrete surface, $r = 90$ deg.

(c) Wood surface, $r = 0$ deg.

(d) Tiles surface, $r = 180$ deg.

(e) Wall cover, $r = 90$ deg.

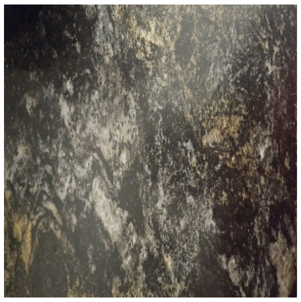
(f) Soiled concrete surface with rocks, $r = 0$ deg.


(g) Tiles surface (other type), with $r = 0$ deg.

(h) Tiles surface (other type), $r = 0$ deg.

(i) Table cover, $r = 270$ deg.

(j) Wood floor, $r = 0$ deg.


(k) Granite surface, $r = 270$ deg.

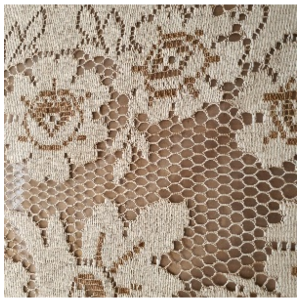

(l) Table cover, $r = 270$ deg.

Fig. 9. Different angles r for images with $i > 0$ deg.

from the ground at which images could be taken may be highly diverse. Two important factors that would dictate the appropriate distances are the resolution of the sensor of the camera and its lenses, which entail the size of each portion of ground represented by each pixel in the image. Hence, in this work, the distance from the ground at which the images were taken were arbitrarily set, as the main goal was to demonstrate the methodology, rather than obtaining a production-level algorithm for a specific mission. Furthermore, the incidence angle does not depend on the distance to the ground at which the images are taken. Therefore, we expect that the features that characterize the angles i and r of an image could be learned by a CNN across multiple images from different distances.

1) Image Pre-processing and augmentation: In general, convolutional neural networks require inputs of a pre-determined size. The size $h = w = 300$ pixels of the input of the CNN was arbitrarily chosen. These values were considered to provide images that were as large as possible (in order to provide the training process with as much information as possible), but still allowing for a training process that could be carried out entirely without being prematurely terminated due to a RAM memory shortage.

Once the images were resized to a common size, every image was rotated clockwise three times (i.e., by 90, 180, and 270 deg), in order to augment the data set. Moreover, as the original images were of RGB type (i.e., their color channels were in the order of red, green, and blue), the set was augmented by converting them to GBR (i.e., green, blue, and red, in that order).

Since the images were RGB (three color channels), each image was represented by a tensor of dimension $300 \times 300 \times 3$. Once the final set of resized and rotated images was completely defined, the intensity value corresponding to each pixel, for each of the color channels, was divided by 255, in order to scale their values into the range $[0,\ 1]$.

III. Results and Discussion

A. Experiment I

In this experiment, both CNN architectures were trained and tested with the same training and testing sets. Recalling the classification categories for Experiment I, Table I and Table II show the confusion matrices obtained with the testing set, for each of the two architectures. In these tables, U indicates an upward inclination, L denotes leftward inclination, D is downward inclination, and R represents rightward inclination.

TABLE I. Confusion Matrix for VGG16 CNN

		Predicted Classes				
		U	L	D	R	N
True Labels	U	43	0	0	3	4
	L	3	40	0	4	3
	D	1	2	41	1	5
	R	2	3	2	39	4
	N	1	1	2	6	40

A few observations can be drawn from Table I and Table II. First, the number of true positive cases for each category, in both tables, strongly suggests that it is feasible to train an algorithm to classify the images, according to the categories defined in this work. This means that, from individual monocular images, CNNs can extract useful information to determine whether the terrain under a landing vehicle is inclined with respect to the attitude of the vehicle, as well as the direction of the inclination.

TABLE II. Confusion Matrix for Xception CNN

		Predicted Classes				
		U	L	D	R	N
True Labels	U	43	1	0	1	5
	L	2	41	1	3	3
	D	1	0	41	0	8
	R	0	4	8	34	4
	N	0	5	3	2	40

TABLE III. Precision and Recall for the VGG16-based CNN

	Precision	Recall	F_1 Score
U	0.86	0.86	0.86
L	0.87	0.80	0.83
D	0.91	0.82	0.86
R	0.74	0.78	0.76
N	0.71	0.80	0.75

TABLE IV. Precision and Recall for the Xception-based CNN

	Precision	Recall	F_1 Score
U	0.93	0.86	0.90
L	0.80	0.82	0.81
D	0.77	0.82	0.80
R	0.85	0.68	0.76
N	0.67	0.80	0.73

We found that the CNN based on VGG16 performed slightly better than its Xception counterpart. Although the numbers were somewhat similar in magnitude, the 39 correctly predicted cases of rightward inclination against the 34, indicated the advantage of VGG16 over Xception. For statistical comparison, we observed the metrics of precision, recall, and F_1 score, as they constitute natural manners to evaluate the performance of classification algorithms. Considering the

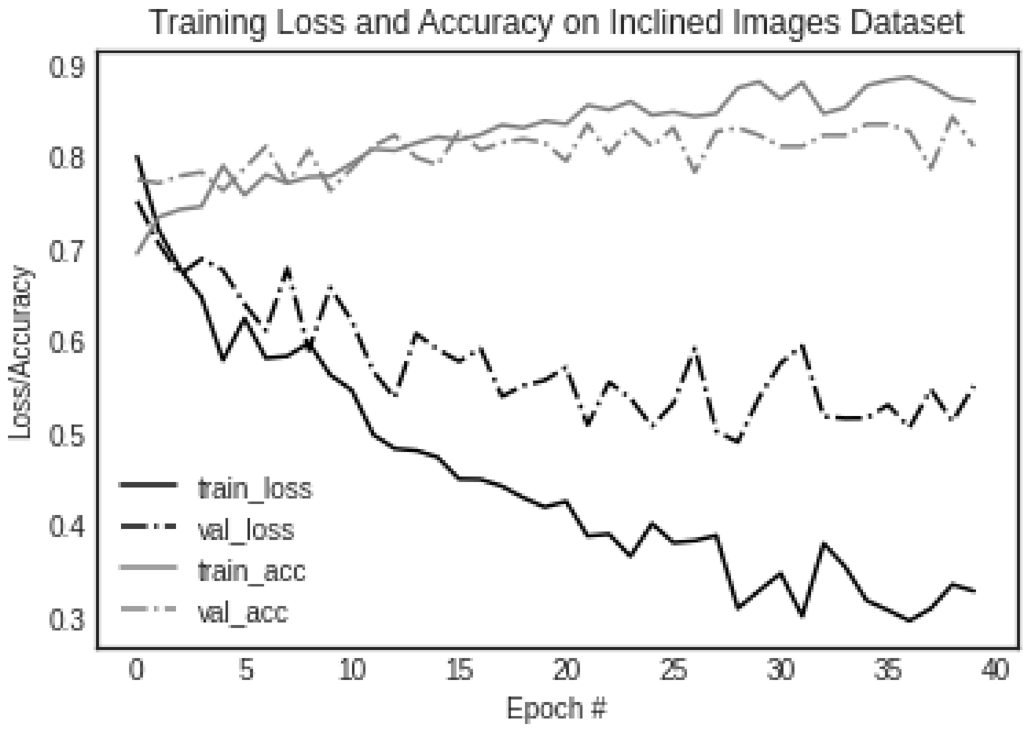


Fig. 10. Evolution of the training of the VGG16-based CNN.

TABLE V. Confusion Matrix: Experiment II

		Predicted Classes								
		N	**20U**	**20R**	**20D**	**20L**	**40U**	**40R**	**40D**	**40L**
True Labels	**N**	220	0	0	0	0	0	0	0	0
	20U	7	34	1	1	0	2	0	0	0
	20R	4	0	49	3	0	0	0	0	0
	20D	9	0	0	40	0	0	0	0	0
	20L	4	4	0	0	38	0	0	0	0
	40U	5	0	0	0	0	50	0	0	0
	40R	4	0	1	0	0	0	38	0	0
	40D	6	0	0	0	0	0	0	54	0
	40L	8	0	0	0	4	0	0	1	43

precision, recall, and F_1 scores displayed in Table III and Table IV, none of the architectures outperformed the other in every other metric.

Fig. 10 illustrates the progress in the prediction capability

TABLE VI. Precision and Recall for Experiment II

	Precision	**Recall**	**F_1 Score**
N	0.82	0.76	0.90
20U	0.89	0.80	0.82
20R	0.96	0.88	0.92
20D	0.91	0.82	0.86
20L	0.90	0.83	0.86
40U	0.96	0.91	0.93
40R	1	0.88	0.94
40D	0.98	0.90	0.94
40L	1	0.77	0.87

of the CNN as the training advanced. It was observed that, after 25 epochs of training, the loss function of the validation set stopped decreasing. We also observed that the loss function computed over the training set and that over the validation set diverged after approximately 10 epochs, which might be indicative of overfitting. This could be resolved by adding more images to the training set.

B. Experiment II

For the performance obtained in Experiment II, using the architecture described in Section II-B1, Table V shows the distribution of classifications obtained for the testing set, while Table VI indicates the precision, recall, and F_1 score statistics obtained. These numbers suggest that the algorithm, indeed, learned to distinguish the classes to which each image belonged. This supports our work towards the next step, which is generating a classifier that can provide estimates of the angles i and r, but within many more classes; thus, describing the measured angles more precisely.

IV. Conclusion

In this work, we explored the feasibility of using convolutional neural networks to evaluate the slope of the terrain under a lander, when it is about to touch ground. This capability may be essential for certain missions, where autonomous landers must accomplish landing maneuvers in unknown terrains.

Two experiments were described, where Experiment II was considered as a natural extension of Experiment I. The latter demonstrated the feasibility of using CNNs to classify image angles into five categories, including normal and four categories with $i > 0$. The former exhibited that it is also possible to quantify the angles i into more than binary categories. The next step will be to train algorithms that can classify the images into many more categories, or treat the problem as a regression one, in which the outputs are real numbers for the angles r and i.

It is important to mention that this work constitutes the first step in a wider project, whose ultimate goal is developing algorithms to precisely estimate angles between landers and the terrain underneath them from monocular images.

Acknowledgment

The researcher(s) would like to thank the Deanship of Scientific Research, Qassim University for funding the publication of this project. In addition, researchers would like to thank the Deanship of Scientific Research, Qassim University for the support of this project.

References

[1] Kulumani, S.; Takami, K.; Lee, T. Geometric control for autonomous landing on asteroid Itokawa using visual localization. Proceedings of the AAS/AIAA Astrodynamics Specialist Conference, Stevenson, Washington, USA, 20–24 August 2017; Univelt, Inc.: Escondido, CA, USA, 2017.

[2] Gaudet, B.; Linares, R.; Furfaro R. Terminal adaptive guidance via reinforcement meta-learning: Applications to autonomous asteroid close-proximity operations. *Acta Astronautica* 2020, *171*, 1–13. https://doi.org/10.1016/j.actaastro.2020.02.036.

[3] Stacey, N.; D'Amico S. Autonomous swarming for simultaneous navigation and asteroid characterization. Proceedings of the AAS/AIAA Astrodynamics Specialist Conference, Snowbird, UT, USA, 19–23 August 2018; Univelt, Inc.: Escondido, CA, USA, 2018.

[4] Sandino, J.; Vanegas, F.; Maire, F.; Caccetta, P.; Sanderson, C.; Gonzalez, F. UAV framework for autonomous onboard navigation and people/object detection in cluttered indoor environments. *Remote Sens* 2020, *12*, 3386. https://doi.org/10.3390/rs12203386.

[5] Albattah, W.; Masood, M.; Javed, A.; et al. Custom CornerNet: A drone-based improved deep learning technique for large-scale multiclass pest localization and classification. *Complex Intell Syst* 2023, *9*, 1299-–1316. https://doi.org/10.1007/s40747-022-00847-x.

[6] Bhaskaran, S.; Nandi, S.; Broschart, S.; Wallace, M.; Cangahuala, L.A.; Olson, C. Small body landing accuracy using in-situ navigation. 2011. Available online: https://trs.jpl.nasa.gov/bitstream/handle/2014/41886/11-0341.pdf?sequence=1&isAllowed=y (accessed on 01 April 2021).

[7] Silva, M.F.; Cerqueira, A.S.; Vidal, V.F.; Honório, L.M.; Santos, M.F.; Oliveira, E.J. Landing area recognition by image applied to an autonomous control landing of VTOL aircraft. Proceedings of the International Carpathian Control Conference (ICCC), Sinaia, Romania, 28-31 May 2017; IEEE: New York, NY, USA, 2017.

[8] Aziz, M.Z.; Mertsching, B. Survivor search with autonomous UGVs using multimodal overt attention. Proceedings of the Safety Security and Rescue Robotics, Bremen, Germany, 26-30 July 2010; IEEE: New York, NY, USA, 2011.

[9] Kawano, I.; Mokuno, M.; Kasai, T.; Suzuki, T. First autonomous rendezvous using relative GPS navigation by ETS-VII. *Navig* 2001, *48*, 49–56. https://doi.org/10.1002/j.2161-4296.2001.tb00227.x.

[10] Furfaro, R.; Cersosimo, D.; Wibben, D.R. Asteroid precision landing via multiple sliding surfaces guidance techniques. *J of Guid, Control, and Dyn* 2013, *36*, 1075–1092. https://doi.org/10.2514/1.58246.

[11] Tognon, M.; Testa, A.; Rossi, E.; Franchi, A. Takeoff and landing on slopes via inclined hovering with a tethered aerial robot. Proceedings of the International Conference on Intelligent Robots and Systems (IROS), Daejeon, South Korea, 09-14 October 2016; IEEE: New York, NY, USA, 2016.

[12] Yano, H.; Kubota, T.; Miyamoto, H.; et al. Touchdown of the Hayabusa spacecraft at the Muses Sea on Itokawa. *Sci* 2006, *312*, 1350–1353. https://doi.org/10.1126/science.1126164.

[13] Liebe, C.C.; Abramovici, A.; Bartman, R.K.; et al. Laser radar for spacecraft guidance applications. Proceedings of the Aerospace Conference (Cat. No.03TH8652), Big Sky, MT, USA, 8-15 March 2003; IEEE: New York, NY, USA, 2003.

[14] Shimkin, P.E.; Baskakov, A.I.; Komarov, A.A.; Ka, M. Safe helicopter landing on unprepared terrain using onboard interferometric radar. *Sensors* 2020, *20*, 2422. https://doi.org/10.3390/s20082422.

[15] Ansar, A.; Cheng, Y. Vision technologies for small body proximity operations. Proceedings of the International Symposium on Artificial Intelligence, Robotics and Automation in Space (i-SAIRAS), Sapporo, Japan, 29 August - 1 September 2010; European Space Agency (ESA): Paris, France, 2010.

[16] Segal, S.; Carmi, A.; Gurfil, P. Stereovision-based estimation of relative dynamics between noncooperative satellites: Theory and experiments. *IEEE Trans on Control Syst Technol* 2014, *22*, 568–584. https://doi.org/10.1109/TCST.2013.2255288.

[17] Sharma, S.; Beierle, C.; D'Amico, S. Pose estimation for non-cooperative spacecraft rendezvous using convolutional neural networks. Proceedings of the Aerospace Conference, Big Sky, MT, USA, 3-10 March 2018; IEEE: New York, NY, USA, 2018.

[18] Herissé, B.; Hamel, T.; Mahony, R.; Russotto, F.X. The landing problem of a VTOL unmanned aerial vehicle on a moving platform using optical flow. Proceedings of the International Conference on Intelligent Robots and Systems, Taipei, Taiwan, USA, 18-22 October 2010; IEEE: New York, NY, USA, 2010.

[19] Herissé, B.; Hamel, T.; Mahony, R.; Russotto, F.X. Landing a VTOL unmanned aerial vehicle on a moving platform using optical flow. *IEEE Trans on Robotics* 2012, *28*, 77–89. https://doi.org/10.1109/TRO.2011.2163435.

[20] Rosa, L.; Hamel, T.; Mahony, R.; Samson, C. Optical-flow based strategies for landing VTOL UAVs in cluttered environments. Proceedings of the World Congress of the International Federation of Automatic Control (IFAC), Cape Town, South Africa, 24-29 August 2014; Elsevier: Amsterdam, Netherlands, 2016.

[21] Green, W.E.; Oh, P.Y. Optic-flow-based collision avoidance. *IEEE Robotics Autom Mag* 2008, *15*, 96–103. https://doi.org/10.1109/MRA.2008.919023.

[22] Beyeler, A.; Zufferey, J.C.; Floreano, D. Vision-based control of near-obstacle flight. *Auton Robots* 2009, *27*, 201–219. https://doi.org/10.1007/s10514-009-9139-6.

[23] Padhy, R.P.; Verma, S.; Ahmad, S.; Choudhury, S.K.; Sa, P.K. Deep neural network for autonomous UAV navigation in indoor corridor environments. *Procedia Comput Sci* 2018, *133*, 643–650. https://doi.org/10.1016/j.procs.2018.07.099.

[24] Lu, Y.; Xue, Z.; Xia, G.S.; Zhang, L. A survey on vision-based UAV navigation. *Geospat Inf Sci* 2018, *21*, 21–32. https://doi.org/10.1080/10095020.2017.1420509.

[25] Mohan, K.; M, P.; S, S.; et al. LIDAR based landing site identification and safety estimation for inter planetary missions. Proceedings of the International Conference on Control, Communication and Computing (ICCC), Thiruvananthapuram, India, 19-21 May 2023; IEEE: New York, NY, USA, 2023.

[26] Xie, H.; Tang, H.; Jin, Y.; et al. An improved surface slope estimation model using space-borne laser altimetric waveform data over the Antarctic ice sheet. *IEEE Geoscience and Remote Sensing Letters* 2022, *19*, 1–5. 10.1109/LGRS.2021.3124224.

[27] Arai, K. Ground control point generation from simulated SAR image derived from digital terrain model and its application to texture feature extraction. *International Journal of Advanced Computer Science and Applications* 2021, *12*. http://dx.doi.org/10.14569/IJACSA.2021.0120112.

[28] Lee, H.Y.; Kim, T.; Park, W.; Lee, H.K. Extraction of digital elevation models from satellite stereo images through stereo matching based on epipolarity and scene geometry. *Image and Vis Comput* 2003, *21*, 789–796. https://doi.org/10.1016/S0262-8856(03)00092-1.

[29] Ajayi, O.G.; Salubi, A.A.; Angbas, A.F.; Odigure, M.G. Generation of accurate digital elevation models from UAV acquired low percentage overlapping images. *Int J of Remote Sens* 2017, *38*, 3113–3134. https://doi.org/10.1080/01431161.2017.1285085.

[30] Krupnik, A. Accuracy assessment of automatically derived digital elevation models from SPOT images. *Photogramm Eng and Remote Sens* 2000, *66*, 1017–1023.

[31] Zhu, X.; Nie, S.; Wang, C.; Xi, X.; Li, D.; Li, G.; Wang, P.; Cao, D.; Yang, X. Estimating terrain slope from ICESat-2 data in forest environments. *Remote Sens* 2020, *12*, 3300. https://doi.org/10.3390/rs12203300.

[32] Stevens, K.A. The information content of texture gradients. *Biological Cybern* 1981, *42*, 95–105. https://doi.org/10.1007/BF00336727.

[33] Ikeuchi, K. Shape from regular patterns. *Artif Intell* 1984, *22*, 49–75. https://doi.org/10.1016/0004-3702(84)90025-0.

[34] Knill, D.C. Surface orientation from texture: Ideal observers, generic observers and the information content of texture cues. *Vis Res* 1998, *38*, 1655–1682. https://doi.org/10.1016/S0042-6989(97)00324-6.

[35] Kanatani, K.; Chou, T.C. Shape from texture: General principle. *Artif Intell* 1989, *38*, 1–48. https://doi.org/10.1016/0004-3702(89)90066-0.

[36] Malik, J.; Rosenholtz, R. Recovering surface curvature and orientation from texture distortion: A least squares algorithm and sensitivity analysis. Proceedings of the European Conference on Computer Vision (ECCV), Stockholm, Sweden, 2-6 May 1994; Springer: Berlin, Germany, 2005.

[37] Malik, J.; Rosenholtz, R. Computing local surface orientation and shape from texture for curved surfaces. *Int J of Comput Vis (IJCV)* 1997, *23*, 149–168. https://doi.org/10.1023/A:1007958829620.

[38] Prince, M.; Alsuhibany, S. A.; Siddiqi, N. A. A step towards the optimal estimation of image orientation. *IEEE Access* 2019, *7*, 185750–185759. https://doi.org/10.1109/ACCESS.2019.2959666.

[39] Romdhani, S.; Vetter, T. Estimating 3D shape and texture using pixel intensity, edges, specular highlights, texture constraints and a prior. Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR), San Diego, CA, USA, 20-25 June 2005; IEEE: New York, NY, USA, 2005.

[40] Saxena, A.; Schulte, J.; Ng, A.Y. Depth estimation using monocular and stereo cues. Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI), Hyderabad, India, 6-12 January 2007; IJCAI: CA, USA, 2007.

[41] Saxena, A.; Chung, S.H.; Ng, A.Y. 3-D depth reconstruction from a single still image. *Int J of Comput Vis (IJCV)* 2008, *76*, 53–69. https://doi.org/10.1007/s11263-007-0071-y.

[42] Handa, A.; Sharma, P. Real-time depth estimation from a monocular moving camera. Proceedings of the International Conference on Contemporary Computing (IC3), Noida, India, 6-8 August 2012; Springer: Berlin, Germany, 2012.

[43] Srikakulapu, V.; Kumar, H.; Gupta, S.; Venkatesh, K.S. Depth estimation from single image using defocus and texture cues. Proceedings of the National Conference on Computer Vision, Pattern Recognition, Image Processing and Graphics (NCVPRIPG), Patna, India, 16-19 December 2015; IEEE: New York, NY, USA, 2016.

[44] Lee, J.; Kim, C. Monocular depth estimation using relative depth maps. Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR), Long Beach, CA, USA, 15-20 June 2019; IEEE: New York, NY, USA, 2020.

[45] Mertan, A.; Duff, D.J.; Unal, G. Relative depth estimation as a ranking problem. *arXiv* 2020, arXiv:2010.06944.

[46] Meingast, M.; Geyer, C.; Sastry, S. Vision based terrain recovery for landing unmanned aerial vehicles. Proceedings of the Conference on Decision and Control (CDC) (IEEE Cat. No.04CH37601), Nassau, Bahamas, 14-17 December 2004; IEEE: New York, NY, USA, 2005.

[47] Rothrock, B.; Kennedy, R.; Cunningham, C.; Papon, J.; Heverly, M.; Ono, M. SPOC: Deep learning-based terrain classification for Mars rover missions. Proceedings of the AIAA Space, Long Beach, CA, USA, 13-16 September 2016; AIAA: Reston, VA, USA, 2016.

[48] Bellone, M.; Messina, A.; Reina, G. A new approach for terrain analysis in mobile robot applications. Proceedings of the International Conference on Mechatronics (ICM), Vicenza, Italy, 27 February - 1 March 2013; IEEE: New York, NY, USA, 2013.

[49] Gonzalez, R.; Iagnemma, K. DeepTerramechanics: Terrain classification and slip estimation for ground robots via deep learning. *arXiv* 2018, arXiv:1806.07379.

[50] Valada, A.; Burgard, W. Deep spatiotemporal models for robust proprioceptive terrain classification. *Int J of Robotics Res* 2017, *36*, 1521–1539. https://doi.org/10.1177/0278364917727062.

[51] Dreiseitl, S.; Ohno-Machado, L. Logistic regression and artificial neural network classification models: A methodology review. *J of Biomed Informatics* 2002, *35*, 352–359. https://doi.org/10.1016/S1532-0464(03)00034-0.

[52] Chaudhuri, B.B.; Bhattacharya, U. Efficient training and improved

performance of multilayer perceptron in pattern classification. *Neurocomputing* 2000, *34*, 11–27. https://doi.org/10.1016/S0925-2312(00)00305-2.

[53] Orhan, U.; Hekim, M.; Ozer, M. EEG signals classification using the K-means clustering and a multilayer perceptron neural network model. *Expert Syst with Appl* 2011, *38*, 13475–13481. https://doi.org/10.1016/j.eswa.2011.04.149.

[54] Al-Saffar, A.A.M.; Tao, H.; Talab, M.A. Review of deep convolution neural network in image classification. Proceedings of the International Conference on Radar, Antenna, Microwave, Electronics, and Telecommunications (ICRAMET), Jakarta, Indonesia, 23-24 October 2017; IEEE: New York, NY, USA, 2018.

[55] Khan, S.; Rahmani, H.; Shah, S.A.A.; Bennamoun, M. *A Guide to Convolutional Neural Networks for Computer Vision*, 1st ed.; Morgan & Claypool: San Rafael, CA, USA, 2018.

[56] Russakovsky, O.; Deng, J.; Su, H.; et al. ImageNet large scale visual recognition challenge. *Int J Comput Vis (IJCV)* 2015, *115*, 211–252. https://doi.org/10.1007/s11263-015-0816-y.

[57] Chen, Y.; Xie, H.; Shin, H. Multi-layer fusion techniques using a CNN for multispectral pedestrian detection. *IET Comput Vis* 2018, *12*, 1179–1187. https://doi.org/10.1049/iet-cvi.2018.5315.

[58] Zhang, S.; Wen, L.; Bian, X.; Lei, Z.; Li, S.Z. Occlusion-aware R-CNN: Detecting pedestrians in a crowd. Proceedings of the European Conference on Computer Vision (ECCV), Munich, Germany, 8-14 September 2018; Springer: Berlin, Germany, 2018.

[59] Goel, T.; Murugan, R.; Mirjalili, S.; Chakrabartty, D.K. OptCoNet: An optimized convolutional neural network for an automatic diagnosis of COVID-19. *Appl Intell* 2021, *51*, 1351–1366. https://doi.org/10.1007/s10489-020-01904-z.

[60] Misra, S.; Jeon, S.; Lee, S.; Managuli, R.; Jang, I.; Kim, C. Multichannel transfer learning of chest X-ray images for screening of COVID-19. *Electron* 2020, *9*, 1388. https://doi.org/10.3390/electronics9091388.

[61] Jain, R.; Gupta, M.; Taneja, S.; Hemanth, D.J. Deep learning based detection and analysis of COVID-19 on chest X-ray images. *Appl Intell* 2021, *51*, 1690–1700. https://doi.org/10.1007/s10489-020-01902-1.

[62] Simonyan, K.; Zisserman A. Very deep convolutional networks for large-scale image recognition. Proceedings of the International Conference on Learning Representations (ICLR), San Diego, CA, USA, 7-9 May 2015; arXiv: New York, NY, USA, 2015.

[63] Tindall, L.; Luong, C.; Saad, A. Plankton classification using VGG16 network. 2017. Available online: https://pdfs.semanticscholar.org/7cb1/a0d0d30b4567b771ad7ae265ab0e935bc41c.pdf (accessed on 01 April 2021).

[64] Ashraf, K.; Wu, B.; Iandola, F.; Moskewicz, M.; Keutzer, K. Shallow networks for high-accuracy road object-detection. *arXiv* 2016, arXiv:1606.01561.

[65] Antony, J.; McGuinness, K.; O'Connor, N.; Moran, K. Quantifying radiographic knee osteoarthritis severity using deep convolutional neural networks. Proceedings of the International Conference on Pattern Recognition (ICPR), Cancun, Mexico, 4-8 December 2016; IEEE: New York, NY, USA, 2016.

[66] Keras. VGG16 and VGG19. 2015. Available online: https://keras.io/api/applications/vgg/#vgg16-function (accessed on 01 April 2021).

[67] TensorFlow. Available online: https://www.tensorflow.org/ (accessed on 01 April 2021).

[68] PyImageSearch. Available online: https://www.pyimagesearch.com (accessed on 01 April 2021).

[69] Chollet, F. Xception: Deep learning with depthwise separable convolutions. Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, USA, 21-26 July 2017; IEEE: New York, NY, USA, 2017.

[70] Keras. Xception. 2017. Available online: https://keras.io/api/applications/xception/ (accessed on 01 April 2021).

[71] Kingma, D.; Ba, J. Adam: A method for stochastic optimization. Proceedings of the International Conference on Learning Representations (ICLR), San Diego, CA, USA, 7-9 May 2015; arXiv: New York, NY, USA, 2015.

[72] Saito, J.; Miyamoto, H.; Nakamura, R.; et al. Detailed images of asteroid 25143 Itokawa from Hayabusa. *Sci* 2006, *312*, 1341–1344. https://doi.org/10.1126/science.1125722.