AN ALGORITHM DEVELOPMENT OF TERRESTRIAL OIL SPILL MODELLING AND OIL VOLUME CALCULATION WITH GIS

Calculation of oil droplet size distribution and surface oil spill modeling: experimental study and Algorithm development/ment

ABSTRACT

The oil spill has very hazardous effects on the environment. The spill caused by oil platforms and oil pipeline networks in water bodies kills thousands of marine creatures because oil pollution in the ocean destroys marine creatures' food sources and makes it hard to breathe and move. Moreover, the oil contamination devastates agricultural areas, and these lands become infertile on terrestrial sites. This type of contamination generally occurs in the same way as water body oil spills. The terrestrial oil spill has a danger of turning into the water-based oil pollution due to oil spill can reach water streams, freshwater sources, and seas. The excellent site of the oil spill is that the trigger point of pollution can be predictable because energy transportation companies know where oil pipelines lie down as petroleum exploration and production companies know the exact coordinates of drilling wells. To minimize the harmful effects of oil contamination on the land site, petroleum companies should prepare scenarios before the disaster happens.

In this study, terrestrial oil spill volume and distribution route on the surface was modeled using Digital Elevation Model (DEM) on Geographic Information System (GIS) technology with its powerful tools on ArcGIS and PostGIS. At the end of the study, the volume of oil that will leak along the pipe route has been calculated according to the topography. Then the oil flow path with puddles location and accumulated oil volume is extracted before a possible oil spill happens. The study will help assess whether sites of oil pipeline valves are selected efficiently or not,

how much area will be affected by pollution, and whether oil reaches freshwater supplies.

Keywords; modeling oil spills on land, terrestrial oil pollution analysis, oil path extraction, GIS

1-INTRODUCTION

Due to the necessity of oil, exploration technology, production sites, and companies' budgets have been expended. For example, Ultra-deep wells can go down to up to 8000 m. depth [1], US field production of crude oil was about 6700 thousand barrels per day in 2008, but it increased to about 16500 thousand barrels per day in 2021 [2]. Five oil and two automobile companies were in the top 10 of the global fortune 500's 2020 list [3].

Whereas negative impacts of petroleum products seriously threaten our future, we might not build our modern civilization without it. After petroleum started to use as an industrial material, we received benefits in various areas such as transportation, electricity, heating, cooling, and clothing [4]. In recent years, our society has been searching for alternative energy sources because environmental pollution and global warming will cause dramatically decrease not only in animals but also in human populations soon. Due to necessity and dependency, sharply getting rid of oil and its products seem impossible in a short period.

Whether or not oil production has been done on a marine site, the petroleum refining process and a significant part of consumption have taken place in the terrestrial site. Most terrestrial oil spills occur during transportation. The leading causes of pipeline oil spills are faulty pipeline welds, landslides, equipment, operator-related causes, terrorist attacks, earthquakes, and corrosions [5, 6]. According to the CIA's factbook, there is more than 300,000 km of the oil pipeline in the world [7]. That means oil pipelines can cover about 7.5 times the world's circumference.

Moreover, the total offshore pipeline length to the onshore pipeline is about 5% because the investment cost of an offshore pipeline is about three times higher and carrying out maintenance and repair activities for an underwater pipeline is harder [8].

Total onshore construction (42,565 miles) beyond 2013 will cost nearly $132 billion	Total offshore construction (2,270 miles) beyond 2013 will cost more than $12 billion
$2.9 billion for 4-10 inc.	$558 million for 4-10 inc.
$19.7 billion for 12-20 inc.	$3.8 billion for 12-20 inc.
$41 billion for 22-30 inc.	$7.9 billion for 22-30 inc.

Table form Oil & Gas Journal[8]

The improving technologies in remote sensing allow detecting hydrocarbon accumulations, indicating visible and invisible oil seepage footprints. Visible-near infrared and short-wave infrared wavelengths can be used for invisible traces of oil seepage, and short and long wave infrared wavelengths can be used for the visible indicator of oil seepage [9-11]. Also, Synthetic Aperture Radar (SAR) can detect visible or invisible oil seepage existence [10-12].

Although terrestrial site oil pollution is at least as dangerous as the oil spill in water, there are many academic sources and risk modeling applications for the oil spill on water bodies [13-16]. Literature reviews show that researchers mainly focus on oil spill pollution's environmental impact on the land [17-19]. Some other researchers focus on how deep the leak can go, depending on the soil types at an incident with predefined or calculated oil volume [20-22]. These studies examined only the vertical distribution of the oil spill and evaluated whether the leakage could reach the aquifers or not. In other studies, GIS was used, researchers mostly modeled actual events, and after the oil spill, the impact of the environmental disaster was analyzed and mapped with actual variables [23, 24]. This study supports the literature for providing necessary tools for identifying possible leaks on the terrestrial site.

A GIS solution will calculate horizontal spill direction and possible spill volume on the ground. There are predefined flow direction and accumulation analysis tools on GRASS [25, 26] and ArcGIS [27, 28]. However, these analyses are not only enough for oil spills because flow direction and accumulation analyses calculate the flow directions of all pixels in the model and the total flood volume for the whole model [29]. Pipeline-induced oil spill starts from a single point and spread

ground from that location. Therefore, a new spill direction approach has been developed for onshore oil spill accumulation, similar to the D8 (eight directions) method on GRASS [25, 26] and ArcGIS [27, 28], having different neighborhood relations. Furthermore, the Algorithm developed via GIS tools for calculating the volume of oil that may leak will be examined in this study.

3. OIL SPILL MODELLING AND VOLUME CALCULATION

Oil spills pose significant environmental and economic risks due to their potential to cause severe damage to natural ecosystems. Various factors contribute to onshore oil spills, with the primary causes being leaks at the drilling point, leaks in oil pipelines and storage tanks, and accidents during oil transportation. These incidents harm the environment, leading to long-term ecological degradation, harm to wildlife populations, and disruptions to local communities that rely on affected ecosystems for their livelihoods [1].

To effectively address these challenges, it is crucial to have accurate information about the location and volume of oil spills. In the United States, the US Environmental Protection Agency (USEPA) plays a pivotal role in registering and monitoring oil spill incidents, while in Europe, the European Environment Agency (EEA) fulfils a similar function. These agencies collect and analyze data on oil spill locations, volumes, and other relevant information, which are crucial for assessing the extent of damage and formulating effective response strategies. By studying historical spill data, patterns can be identified, enabling policymakers, industry professionals, and environmentalists to develop targeted measures to mitigate future spills.

In addition to the USEPA and EEA, organizations such as the Pipeline and Hazardous Materials Safety Administration (PHMSA) in the USA and the European Gas Pipeline Incident Data Group (EGIG) in Europe specifically record and analyze pipeline spill data. These organizations are vital in monitoring pipeline integrity, identifying vulnerabilities, and implementing measures to

prevent leaks and spills. By gathering data on pipeline incidents, including causes and locations, these organizations contribute to a better understanding of the factors determining to onshore oil spills.

Statistical information compiled by these agencies reveals that pipelines significantly cause both onshore and offshore oil spills. Pipeline failures can occur for various reasons, including corrosion, material defects, poor maintenance, or external factors such as excavation damage [2]. The risks associated with pipeline transportation necessitate developing and implementing stringent inspection and maintenance programs to detect potential weaknesses and address them promptly. Regular inspections, adherence to industry standards, and advanced technologies such as inline inspection tools and acoustic monitoring systems can help identify and mitigate potential pipeline failures before they result in spills.

Furthermore, oil storage tanks represent another significant cause of onshore oil spills. These tanks, which store large quantities of oil for various purposes, can develop leaks due to structural weaknesses, inadequate maintenance, or improper handling [3]. Effective storage tank management practices, including regular inspections, proper maintenance, and secondary containment systems, are crucial to prevent spills. Additionally, implementing advanced technologies such as automated leak detection systems can provide early warning signals and enable swift response in case of tank failures.

Although statistical information and records can be used to analyze the causes and consequences of oil spills, taking proactive measures before these events occur is paramount. Geographic Information System (GIS) analysis can play a significant role in achieving this. GIS integrates geographical data with analytical capabilities, allowing for identifying vulnerable areas and predicting potential spill scenarios based on topographic variables. GIS analysis can provide insights into the possible spread and distribution of oil leakage in horizontal and vertical directions by considering terrain features, elevation, and proximity to

environmentally sensitive areas. This information enables stakeholders to prioritize prevention efforts, target inspections, and maintenance activities, and develop effective emergency response plans tailored to specific locations and potential spill scenarios.

Moreover, integrating GIS technology with predictive modeling techniques allows for a more accurate assessment of the potential volume of oil that could leak from pipeline cracks. By incorporating factors such as pipeline characteristics, flow rates, and environmental conditions, predictive models can estimate the potential volume of oil release and its dispersion in the surrounding areas. This knowledge aids in developing contingency plans, resource allocation, and implementing measures to minimize the environmental impact of oil spills.

Today, UAV, satellite and photogrammetric images are used along with GIS to detect the possible pipeline leak route [4]. With the help of these images, the situation of the land before the leak and the situation after the leak can be compared. This makes it easier to apply the algorithms developed for leak detection to the field in terms of both route accuracy and sensitivity.

In summary, addressing the causes and consequences of onshore oil spills requires a multi-faceted approach encompassing accurate data collection, rigorous inspections, and applying advanced technologies such as GIS analysis, images and predictive modeling. By understanding the factors contributing to oil spills and their potential impacts, stakeholders can implement targeted preventive measures to minimize the occurrence and mitigate the consequences of such incidents. GIS analysis allows for assessing vulnerable areas and predicting spill scenarios, enabling informed decision-making and effective emergency response planning. Ultimately, these efforts contribute to protecting the environment, preserving natural resources, and the well-being of communities affected by oil spills.

3.1. Supervisory Control and Data Acquisition (SCADA)

Supervisory Control and Data Acquisition (SCADA) is crucial in modern industrial control systems. It facilitates the monitoring, controlling, and real-time data collection via the control center for these systems. It is a vital intelligence gathering, control, and supervision node for diverse industrial operations. With technological advancements, machine learning and artificial intelligence can be implemented to analyze the collected data, improving system efficiency and effectiveness [5, 6].

One salient application of SCADA can be found within pipeline systems, where it brings several remarkable capabilities. To illustrate, it provides an automated start-up and shutdown functionality. This is significant for operations involving equipment such as pumps, where SCADA systems can autonomously initiate or cease their operation based on various system parameters or conditions.

In addition to this, SCADA systems are equipped with a mechanism for an emergency shutdown. This functionality activates when the pipeline pressure falls critically low, or leakage is detected. The system can rapidly respond to potential hazards through such mechanisms, thereby maintaining system integrity and minimizing potential environmental and financial damage.

Moreover, the SCADA system aids in configuring the pipeline infrastructure, which includes the operations of valves - whether they need to be opened or closed. This function offers fine control over the pipeline operations, making the system adaptive and responsive to varying operational needs.

Another noteworthy feature provided by SCADA systems is real-time modeling. This functionality assists in eliminating any erroneous readings by simultaneously examining multiple data sets. By identifying and excluding inaccurate data, real-time modeling helps maintain the credibility and accuracy of the system's data, thus enhancing decision-making processes.

Furthermore, SCADA can detect leaks by monitoring for severe pressure and flow rate drops. Upon detecting such a drop, the system can pinpoint the location of the leakage and initiate preventive measures. This capability aids in timely leak detection and remediation, mitigating any substantial impact on system performance and the environment.

Batch tracking represents another major functionality provided by SCADA. It enables the separation of different shipments at the destination point, ensuring accurate tracking and allocation of resources. This function, therefore, brings a significant degree of efficiency and accountability to the overall pipeline operation.

A function related to accuracy and validation in SCADA systems is meter proving. Here, various parameters, such as the transported product's pressure, temperature, density, and flow rate, can be compared at many different points. This process ensures the accuracy and reliability of meter readings, contributing to the overall system performance and data credibility.

Regarding communication, SCADA systems utilize diverse methods such as copper cables, radio links, GPRS modems, and fiber cables. Modern systems often incorporate more than one of these methods with redundancy, ensuring a reliable and consistent data exchange between the SCADA system and the equipment.

Upon detecting a leak in an oil pipeline, SCADA systems are programmed to automatically shut down the pump systems and valves before and after the leakage site. By doing so, the system minimizes the leakage volume and ensures the safety and efficiency of pipeline operations, ultimately showcasing the comprehensive functionality of SCADA in industrial control systems.

3.2. Oil Spill Volume Calculation

Despite high-speed fiber connections linking the SCADA system and pipeline equipment, the flow rate will be limited by the time between leak detection and valve closure. Furthermore, an additional flow can occur due to trapped oil, resulting from establishing atmospheric pressure equilibrium.

3.2.1. Calculation of Leakage Volume from Oil Storage Tanks

The likelihood of onshore oil spills resulting from in-ground oil storage tank rupture is minimal. The primary causes of tank rupture are lightning, maintenance or hot work activities, operational errors, equipment failures, and intentional sabotage [7]. The primary concern associated with oil storage tanks is the potential for explosions. To mitigate this risk, these tanks are designed with floating roofs to prevent oil evaporation and reduce the likelihood of an explosion. However, it is important to note that even with such measures in place, the risk of explosion cannot be eliminated. Therefore, further preventive and mitigation measures must be implemented to minimize tank rupture risk [8]. These measures include regular inspections and maintenance, installing lightning protection systems, utilizing fire detectors, and implementing tank cooling systems. By employing these strategies, the risk of rupture in storage tanks can be significantly reduced. If a rupture occurs in an oil storage tank, the volume of oil above the rupture point may escape. The calculation of volume to flow in a regularly shaped storage tank depends on parameters such as the height of the tank (h), the rupture height from the tank bottom (l), and the base area of the tank (S).

The volume of oil that will flow from the oil tank if no measures are taken

$$V = (h - l).S \qquad [m^3]$$

If the volume of the round, irregularly shaped storage tank is

$$V = \pi \int_{h-l}^{h} [f(x)]^2 dx \qquad [m^3]$$

3.2.2. Calculation of Leakage Volume from Oil Pipelines

The total leakage volume refers to the quantity of fluid that escapes between a damaged section of a pipe and the point at which a valve is fully closed. It encompasses the cumulative volume of fluid that flows until the pressure within the system is balanced after the valves have been shut, assuming no corrective actions are taken in oil pipelines.

3.2.2.1. Leakage Volume in Time until Valves Shut-in

The US Department of the Interior, specifically the Bureau of Ocean Energy Management, has proposed two distinct approaches for estimating the leakage volume resulting from a rupture in a pipeline [9]. It is important to note that these methods are not applicable for assessing minor fractures or pinhole leaks in pipelines.

- IDpipe [in]: Pipeline internal Diameter
- Lpipe [ft]: Pipeline length
- $X_{ovf}^{initial}[\text{Dimensionless}]$: Oil volume fraction at ambient pressure and temperature
- $X_{ovf}^{amb}[\text{Dimensionless}]$: Oil and gas densities at ambient pressure and temperature
- $\rho_o^{initial} and \rho_g^{initial}$ [lb/ft³]: Oil and gas densities at initial operational conditions
- $\rho_o^{amb} and \rho_g^{amb}$ [lb/ft³]: Oil and gas densities at ambient pressure and temperature
- GOR [scf/stb]: Gas-oil ratio at standard conditions in pipeline
- ρ_L^{stc} [lb/ft³] and γo [-] (specific gravity, dimensionless): Oil density at the standard condition in the pipeline
- ρ_g^{stc} [lb/ft³], γg [-] (Specific gravity, dimensionless): Pipeline gas density at the standard condition
- Q [stb/d]: Pipeline flow rate
- t [min]: Time until valve close

The basic formula is:

$$V_{Pre-Shut} = \frac{Q.t}{1140} \quad \text{[bbls]}$$

The advanced formula is:

Calculating of total volume of pipe

$$V_{pipe} = \left(\frac{ID_{Pipe}}{24}\right)^2 . \pi . L_{pipe} \qquad [ft^3]$$

Calculating the initial mass

$$m_{tot}^{initial} = \left(\rho_o^{initial} . V_{pipe} . X_{ovf}^{initial}\right) + \left(\rho_g^{initial} . V_{pipe} . \left(1 - X_{ovf}^{initial}\right)\right) \qquad [lb]$$

Calculating the total mass

$$m_{tot}^{amb} = \left(\rho_o^{amb} . V_{pipe} . X_{ovf}^{amb}\right) + \left(\rho_g^{amb} . V_{pipe} . \left(1 - X_{ovf}^{amb}\right)\right) \qquad [lb]$$

Calculating the total mass released

$$m_{rel} = m_{tot}^{initial} - m_{tot}^{amb} \qquad [lb]$$

Calculating the gas mass fraction at standard conditions

$$X_{gmf}^{stc} = \frac{1}{1 + \frac{\rho_L^{stc} . 5.614583}{(GOR . \rho_g^{stc})}} \qquad [-]$$

Calculate the volume of oil releases. Also, this is the basic formula.

$$V_{Pre-Shut} = \frac{Q.t}{1140} \qquad [bbls]$$

Calculating the volume of oil released. This equation has taken from the advanced formula.

$$V_{rel}^{stc} = 0.1781 . \frac{m_{rel}\left(1 - X_{gmf}^{stc}\right)}{\rho_o^{stc}} + V_{Pre-Shut} \qquad [bbls]$$

3.2.2.2. Volume of Flow That May Occur after Valves Shut-In

Following the detection of a leak in oil pipelines, valves are promptly closed to mitigate environmental pollution. Without any intervention, the flow of oil will persist until the internal pressure of the pipe aligns with the external pressure, as previously discussed. The total volume of oil confined between two valves can be determined using the formula

$$V_{pipe} = \left(\frac{ID_{Pipe}}{24}\right)^2 . \pi . L_{pipe} \qquad [ft^3]$$

However, it is essential to note that not all of this volume will spill. The specific volume of oil that will be discharged from the trapped space between two valves can be calculated through GIS analysis.

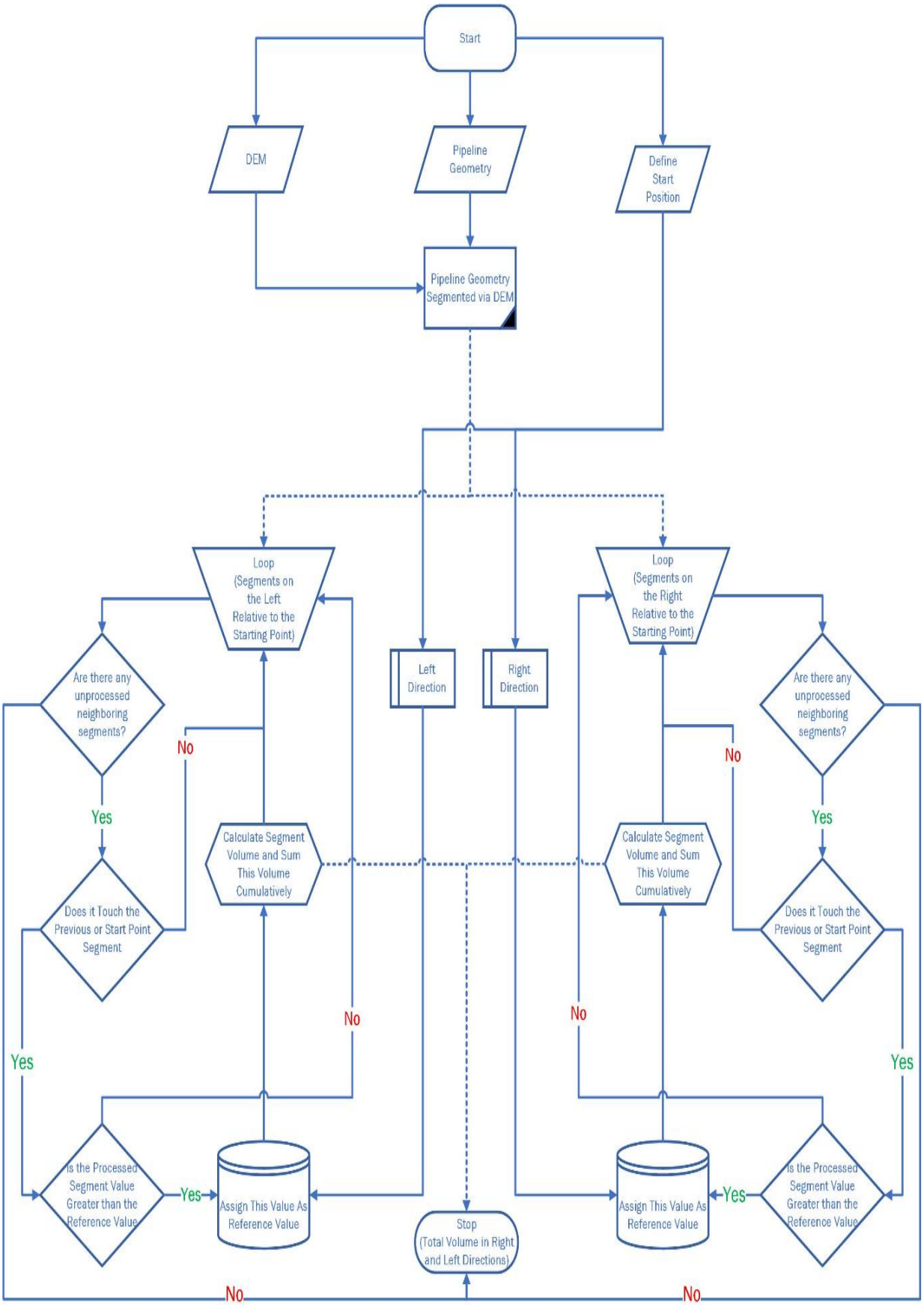


Figure 1. Flow Chart of Pipeline Spill Value Calculation

A Python-based algorithm utilizing a PostgreSQL database and GIS has been developed to analyze the volume of oil that will be discharged after the closure of valves. The algorithm's working principle can be comprehensively understood by reviewing its step-by-step process, as depicted in Figure 1.

Step 1: Input data supply

DEM data of the target region, pipeline vector drawing with coordinates, and leak location coordinates on the pipeline are provided as input.

Step 2: Overlapping Pipeline Geometry and DEM Data

The pipeline geometry and DEM data are overlapped to identify the areas where they coincide.

This overlapping process helps segment the pipeline based on the DEM data.

Step 3: Intersection with Starting Point and Reference Value Recording

The DEM data is intersected with the starting point of the pipeline.

A randomly selected segment, either in the right or left direction, is designated as the processed segment.

The height value at the intersection point is recorded as the reference value.

Step 4: Calculation of the Volume for the Selected Segment

The volume of the selected pipeline segment is calculated.

This volume is recorded as the total leakage volume.

The selected segment is marked as processed to avoid redundancy.

Step 5: Loop for Segment Inspection

The algorithm enters a loop to inspect all parts of the pipeline route segmented with DEM data.

This loop iterates through the following sub-steps:

a. Sub-step 5a: Check if Segment Has Been Processed

The algorithm checks whether the selected segment has been processed before.

If it has not been processed, a new segment is randomly selected, and the loop restarts from Step 5.

If it has been processed, the algorithm proceeds to the next sub-step.

b. Sub-step 5b: Geometric Touch Check

The algorithm checks whether the loop segment geometrically touches the processed segment.

If there is no touch, a new unprocessed segment is randomly selected, and the loop restarts from Step 5.

If there is a touch, the algorithm moves to the next sub-step.

c. Sub-step 5c: Path Update and Volume Calculation

Within each iteration of the loop, the algorithm updates the path based on the height value of the processed segment.

If the height value of the processed segment is smaller than the reference height value, the path is updated.

If the height value is greater than or equal to the reference value, the height of the processed segment becomes the new reference height value.

The volume of the processed segment is calculated, and the total leakage volume value is added to this volume.

The newly calculated volume is recorded as the total leakage volume.

The processed segment is marked as selected, and the loop restarts with the newly selected segment.

Step 6: Switching to Unselected Direction

After inspecting all segments in one direction, the algorithm switches to the unselected direction (right or left).

The height value of the intersection point between the DEM data and the starting point is reassigned as the reference value.

Step 7: Repeating Step 5 for the New Direction

The loop described in Step 5 is repeated for the newly selected direction.

The algorithm goes through the same sub-steps described in Step 5 for the new direction.

By following these steps, the GIS-based algorithm, in conjunction with the PostgreSQL database and Python programming language, determines the volume of oil that will flow after the valves are closed. The algorithm considers the DEM data, pipeline geometry, and the coordinates of the leak location to estimate the total leakage volume accurately.

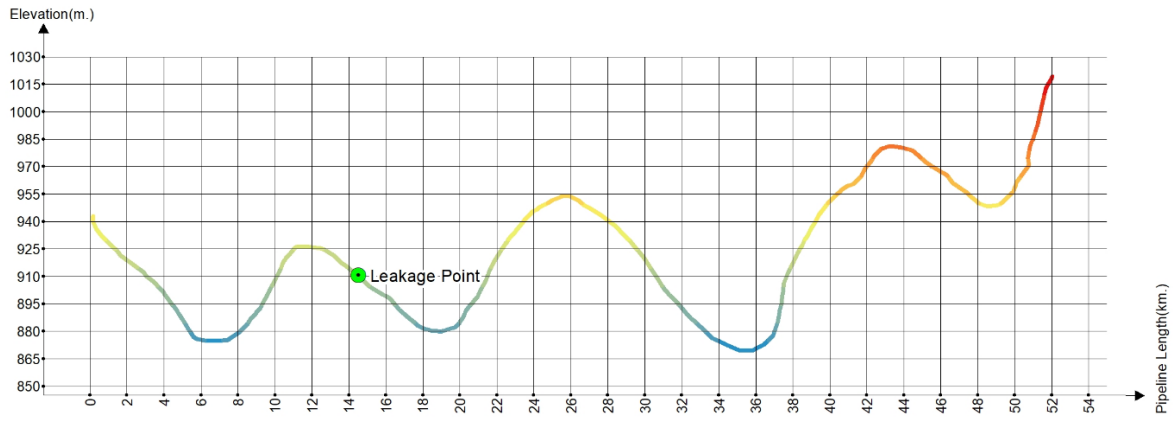


Figure 2. How Pipeline Spill Value Calculate (color shows height).

The developed Algorithm has been applied to a pipeline segment whose elongation section representation is like Figure 2.

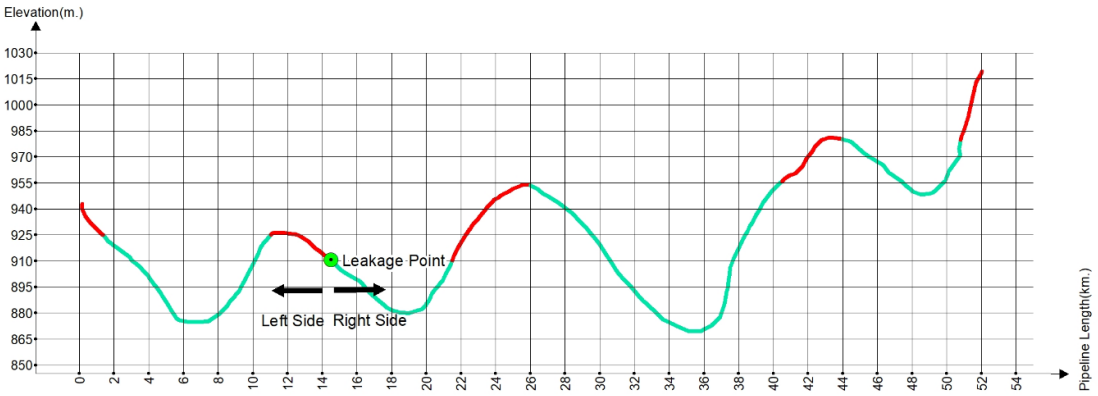


Figure 3. The Result of Pipeline Spill Value in Chart

The application yielded anticipated and satisfactory outcomes when executed in both the right and left directions, starting from the specified leak point. In Figure 3, the regions highlighted in red represent areas where oil is projected to be discharged. Conversely, the green-colored regions indicate the oil segments that remain due to the influence of the topographical features, as depicted in Figure 3.

3.3. Horizontal Distribution Of Oil Spill On Surface

Several widely recognized hydrology analyses based on Geographic Information Systems (GIS) exist, including flow direction, flow accumulation, and sink detection algorithms. While these analysis methods may appear relevant to onshore oil spills based on their names, they are not directly applicable to such scenarios. These algorithms typically operate by uniformly evaluating equal amounts of water drops across each Digital Elevation Model (DEM) cell and analyzing all cells. They are commonly utilized for flood modeling or extracting topographical features, such as ridges, stream channels, or depressions [10].

In the context of terrestrial oil spills, they typically occur due to pipeline cracks or unintentional accidents on drilling rigs, resulting in a gradual spread of the spilled liquid across the affected area. This process involves gradually releasing an exact

quantity of oil, and progressively dispersing it into the surrounding ground from a specific location. Consequently, the application of conventional hydrological algorithms, as previously mentioned, proves inadequate in addressing the unique characteristics of an oil spill.

The primary criterion for accurately determining the route of an oil spill lies in identifying the presence of depressions or holes in the flow path. In such instances, the spilled liquid accumulates until its height aligns with that of the depression. Generally, oil flows downhill from higher to lower elevations. Consequently, when employing a GIS-based solution to simulate an oil spill, the liquid consistently follows the path of the lowest neighboring cell. However, if all neighboring cells possess higher elevations than the current cell, the oil accumulates to a height equivalent to that of the lowest neighboring cell before proceeding further.

The soil's absorption rate is another critical factor influencing the length and extent of the oil spill's flow path. Various soil types exhibit distinct absorption capacities, determining the amount of liquid absorbed. In a GIS-based analysis, it is possible to define the absorption rate of the liquid based on the specific soil type within a given unit area. This information helps refine the simulation model, providing a more accurate representation of the oil spill's behavior.

Additionally, it is important to consider various factors contributing to reducing the spilled oil volume. These factors may include evaporation, attachment of oil to vegetation within the soil, small cracks in the ground, and other similar phenomena. These losses should be considered to provide a more realistic estimation of the oil spill's impact.

The oil spill analysis algorithm should be designed to terminate when the cumulative losses along the spill path exceed the initially considered volume of

oil. This ensures the simulation accurately reflects the real-world scenario and prevents overestimating potential impacts.

By considering these specific parameters and incorporating them into a comprehensive GIS-based analysis, it becomes possible to simulate the path and behavior of an oil spill, thereby facilitating more accurate assessments of the spill's potential impacts on the surrounding environment and enabling the development of effective response strategies.

Oil spill analysis was coded using the arcpy library on ArcGIS by following the steps below (Figure 4).

1. In the initial stage of this analysis, various inputs are fed into the application. These inputs consist of Digital Elevation Model (DEM) data, information on the volume of the oil that has leaked, a map detailing different soil types in the area, and where the leak started. These inputs serve as a basis for the algorithm to conduct its operations.

2. The second step involves integrating the DEM data and the soil type map to create a surface model. This model illustrates the terrain's elevation and the type of soil found at each location. This is achieved by conducting an intersection analysis within the Geographical Information System (GIS), which overlays the two data types to create a combined visual representation. The rectangular pixels may become somewhat distorted at this stage due to the complex interactions between different data layers. However, such distortion is considered insignificant for this algorithm.

3. Following this, an intersection analysis is performed between the leak's starting point and the DEM data within the GIS. This process identifies the pixel where the leak starts and tracks its path on the model. The elevation value of this pixel is then noted down and set as the reference height for further steps in the algorithm.

4. The algorithm then runs in a loop until the total volume of the leak, calculated by the system, meets or exceeds the leaked volume input at the beginning of the process. The various steps within this loop are as follows:

a. The algorithm first identifies the pixel with the smallest elevation value along the path of the leak. The elevation value of this pixel is then set as the reference height for the next stage of the algorithm.

b. If other pixels along the leak path have an elevation equal to the reference height, they are marked as the reference path. The algorithm then goes into an iterative process, examining the surrounding pixels and adding any with equal height to the reference path. This is performed using an endless loop or a recursive function, a coding technique that allows a function to call itself. This process is akin to a breadth-first search algorithm, a strategy for searching in a graph when breadth (neighbors to a node) is prioritized before depth (children of a node).

c. At this stage, the algorithm calculates the volume of oil accumulated in pixels with a height less than the reference height along the leak path. This is computed by multiplying the pixel area being processed by the difference between the reference height and the height of the pixel in question. The formula can be written as:

Pixel-based Puddle Volume = Area of the Processed Pixel * (Height of Reference Pixel - Height of the Processed Pixel)

d. The algorithm then calculates the amount of oil absorbed by each section of the DEM, based on the unit volume absorption amount for the specific type of soil in that location.

e. At this point, the algorithm accounts for additional losses due to small surface cracks, evaporation, and oil adhering to vegetation. This is done by defining a constant loss parameter proportional to the calculated path length. The total losses are then calculated by multiplying the path length by the constant loss parameter.

f. The loop is terminated as soon as the combined volumes of oil lost in steps c, d, and e meet or exceed the total volume of the oil leak as given in the input parameters.

In the developed Algorithm, the soil type is an optional parameter. A fixed absorption rate is given if this data does not exist.

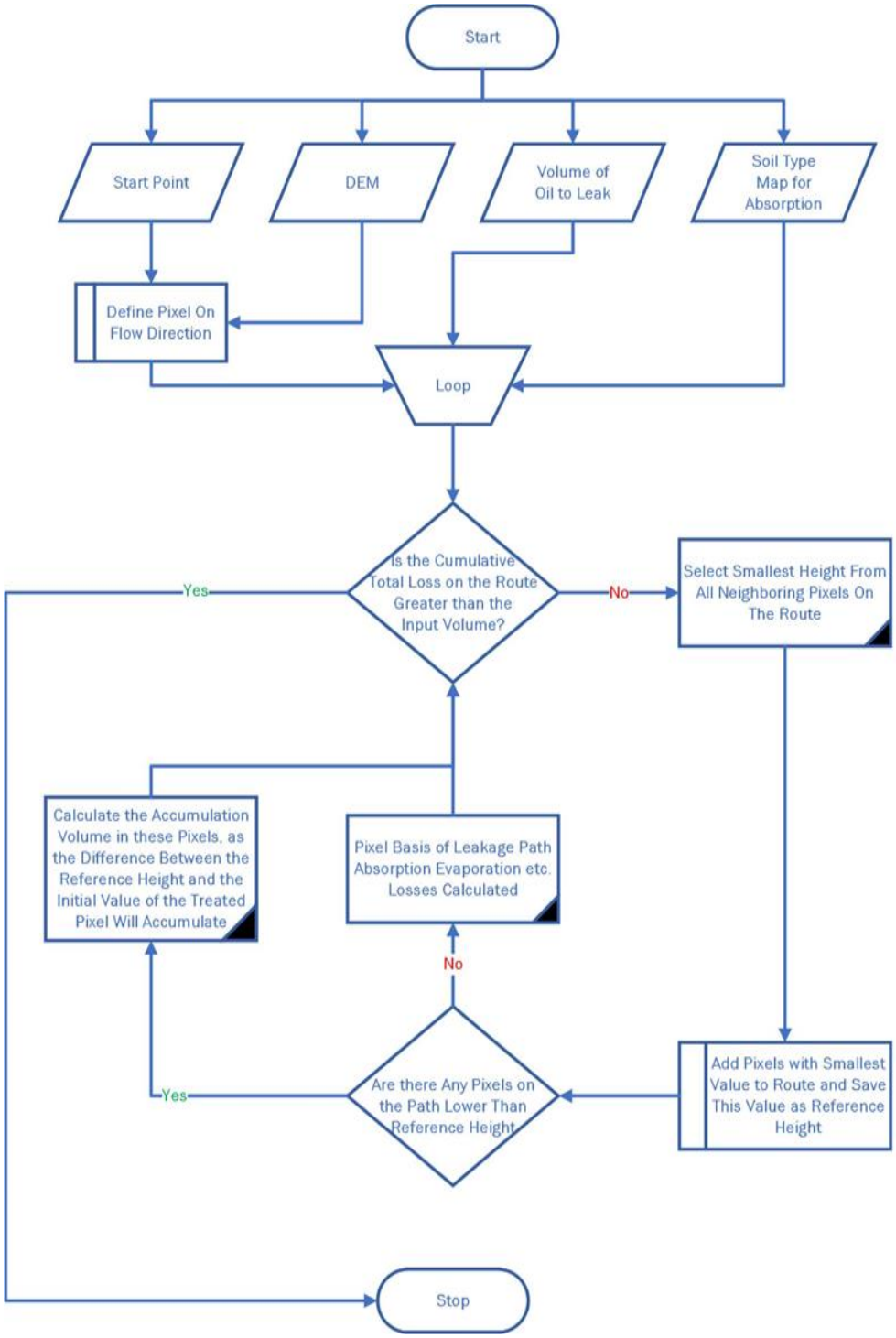


Figure 4. The Algorithm of Horizontal Oil Spill Distribution

3.3.1. Lowest Neighbor Problem

In the realm of raster-based analysis, there are fundamentally two ways in which we define a pixel's neighbors, the '4-pixel neighborhood relation' and the '8-pixel neighborhood relation'.

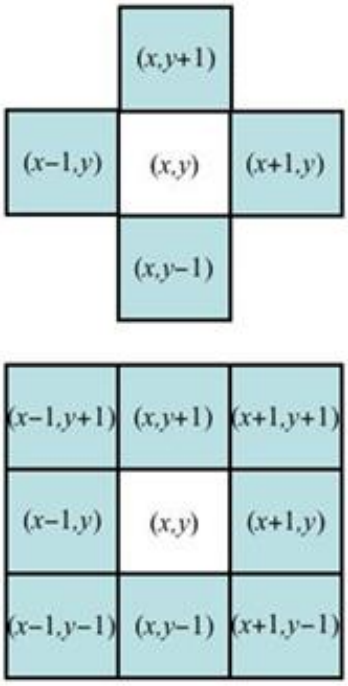


Figure 5. Pixel Neighborhoods

When explain to the 4-pixel neighborhood relation, In this case, each pixel is viewed in relation to its four immediate neighbors, situated in the cardinal directions of North, South, East, and West. If single pixel with coordinates (x, y) are taken into consideration, its neighbors would be found at the coordinates (x, y+1) for the pixel to the North, (x, y-1) for the pixel to the South, (x+1, y) for the pixel to the East, and finally, (x-1, y) for the pixel to the West (top image in Figure 5). Only the pixels that share an edge with the central pixel are considered neighbors in this configuration.

On the other hand, the 8-pixel neighborhood relation expands the concept of a pixel's neighbors to include those pixels that share a vertex with the central pixel. Those pixels situated diagonally to the central pixel must also be considerated.

In addition to the four coordinates detailed in the 4-pixel neighborhood relation, the neighbors would also include (x+1, y+1) for the pixel to the Northeast, (x+1, y-1) for the pixel to the Southeast, (x-1, y-1) for the pixel to the Southwest, and (x-1, y+1) for the pixel to the Northwest (bottom image in Figure 5).

In the realm of oil spill analytical studies, the traditional understanding of neighboring pixels undergoes a transformation. Within this domain, our focus isn't solely on stationary pixels. Instead, we delve into pixels that craft an ever-evolving and enlarging trajectory. As the oil spill proliferates, so does the trajectory, and concomitantly, the neighboring pixels amplify in quantity, undergoing dynamic modifications as the trajectory integrates novel pixels (Figure 6).

Should one approach the study of oil spill analytics through the conventional lens of raster-oriented trajectory discernment, it would necessitate a perpetual documentation of coordinates pertaining to all adjacent pixels. With each inclusion of a fresh pixel into the spill trajectory, such documentation demands revisions to reflect the evolving vicinity of the impacted pixels. Such an approach would undeniably introduce heightened computational intricacy due to the perpetually mutable data.

An optimized methodology can be employed to circumvent this computational intricacy: transmuting Digital Elevation Model (DEM) data into a vectorial format, including their altitudinal metrics. The vectorial data paradigm, employing vertices, lines, and polygons for terrestrial representation, typically offers enhanced adaptability and is adept at encapsulating intricate geospatial attributes with heightened accuracy compared to its raster counterpart. This metamorphosis ensures the retention of pivotal data for oil spill analytics without the computational challenges inherent in incessant updates of adjacent pixel data.

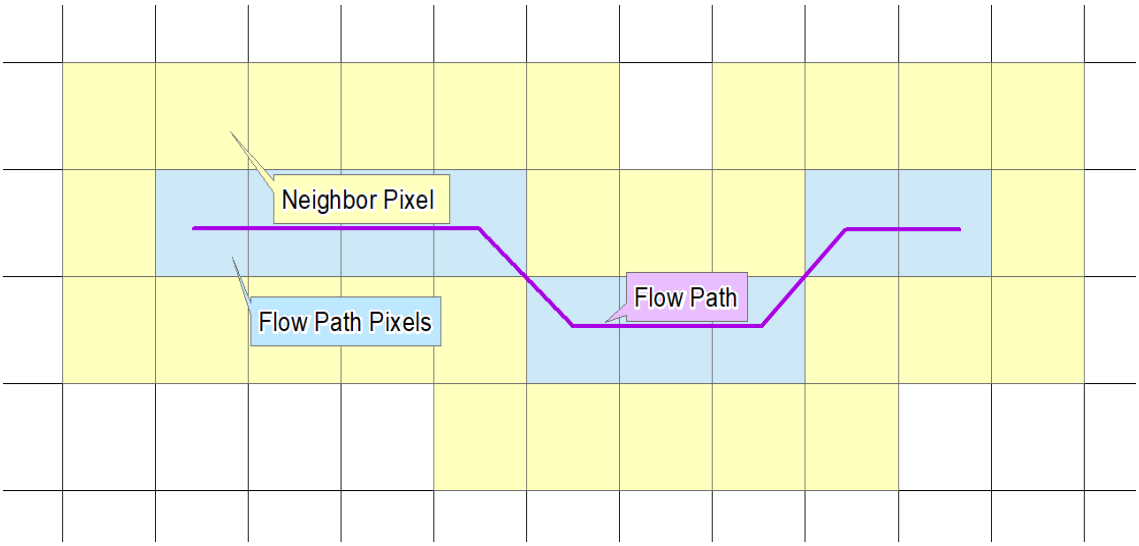


Figure 6. Neighbor Relation in Horizontal Oil Spill Distribution

3.3.2. Oil Puddle Problem

In addressing the critical issue of oil spills, there is a key parameter to be identified and understood—'barrier pixels'. In the context of an oil spill, 'barrier pixels' refer to specific digital markers on the spill's path. Each spill region is represented as a pixelated grid, with each pixel given a numerical value that signifies its 'height' or relative elevation.

A unique trait that distinguishes barrier pixels is their role in fluid accumulation. Drawing parallels with a dam or blockade, these barrier pixels retain the oil spill until such a point when the 'height' of the leaked oil behind them matches their own height value. At this juncture, just like a dam at capacity, the oil begins to overflow and continues along its path.

In fluid accumulation, the height of the neighboring pixels plays a crucial role. This procedure particularly involves the pixel with the smallest height value among the neighbors. Fluid builds up when this pixel's height is larger than the previously processed pixel's height. The quantity of this buildup is ascertained by the difference in height values between the barrier pixel and the pixel processed

before it. This difference measures the amount of fluid accumulating before reaching the barrier pixel's 'capacity' and spilling over.

As the fluid, or oil, permeates through the grid, each pixel on the predicted route carefully inspects its height value. Intriguingly, fluid accumulation will occur in every pixel with a height less than that of the last identified barrier pixel. This behavior is analogous to the natural fluid movement across varying elevations, which flows from higher to lower regions until a barrier is surpassed.

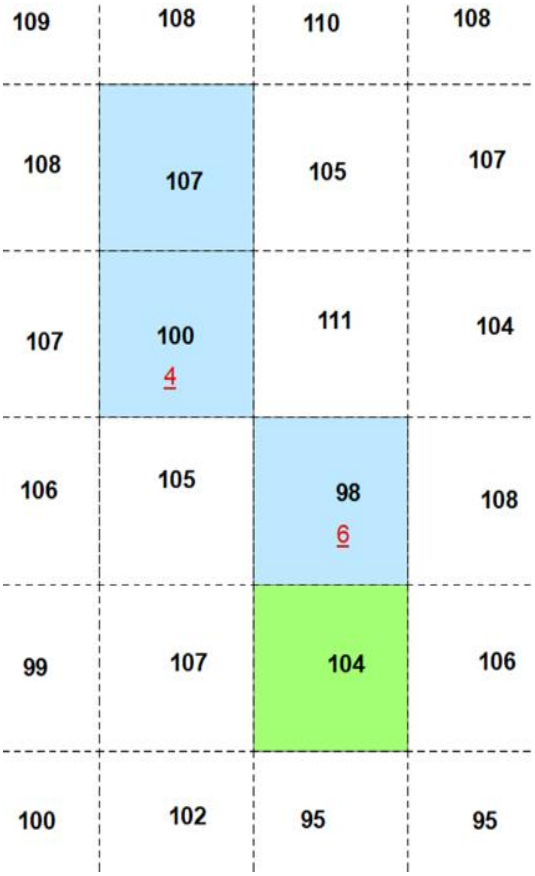


Figure 7. Oil Puddle Problem

To illustrate this, consider an image-based scenario. After processing a pixel with a value of 98, it is found that all the neighboring pixels have a higher value than 98. This scenario signals the commencement of fluid accumulation up until the height of the smallest neighboring pixel. In this instance, the smallest neighboring pixel has a height of 104. Consequently, the pixel with a height of 98 will accumulate 6 meters of fluid, computed by subtracting 98 from 104. Similarly, a pixel with a height of 100 will accumulate 4 meters of fluid, representing the difference between 104 and 100 (Figure 7).

In essence, understanding the behavior of barrier pixels and the dynamics of fluid accumulation is an essential part of mitigating oil spills. This detailed explanation enhances comprehension of the mechanisms at play in this context, supporting effective strategies to manage and reduce the environmental impacts of oil spills.

3.3.3. Neighbor's Neighbor Problem

In real-world scenarios, liquid, like water or oil, typically flows along a gradient from higher elevations to lower ones. This principle is echoed in Geographic Information Systems (GIS), where the motion of liquid is modelled from higher to lower cells within a grid representing the terrain. An essential factor to consider while calculating this flow direction is the treatment of flat cells within the Digital Elevation Model (DEM), which represents the surface of the Earth in a digital format.

In the DEM, cells are essentially pixels assigned specific elevation values. In cases where multiple cells within the neighborhood share the same elevation, the fluid is modeled to traverse these flat cells. This scenario resembles how water would flow over a flat surface in the physical world, distributing itself evenly across the plane.

Our analytical process necessitates a further step after identifying the lowest pixels on the fluid's projected path. Neighbouring cells with the same elevation values as these lowest path pixels must be examined. This inspection aims to determine whether these equal-elevation cells are adjacent or connected to the flow path pixels, implying that they form part of the boundary of the fluid's path.

This step serves to extend the currently calculated flow path. It acknowledges that fluid won't necessarily follow a strict path of descending elevations, but might also spread laterally across areas of the same height. By doing so, we ensure

that our model accurately reflects the potential for fluid to spread across equal elevations.

Let's examine an illustrative example to clarify this concept further. Consider an image that represents the flow of a liquid from an area with an elevation value of 107 to an area with a value of 94. A pixel with 96 (represented in green) has been processed on this image. The next step in our application would be to designate cells with an elevation value of 95 (shown in yellow) as part of the projected flow path.

However, our task does not end here. All neighboring cells of these 95-value cells must also be examined to determine if any other cells have an equal elevation value of 95. These cells form part of the flow path since the fluid can spread to these areas. This way, the flow path is expanded to include these same-elevation cells, providing a more comprehensive understanding of the fluid's trajectory (Figure 8).

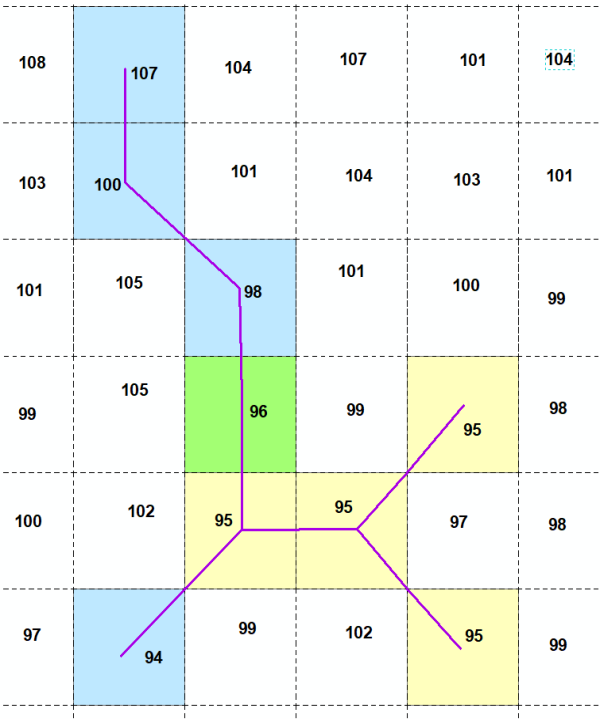


Figure 8. Neighbor's Neighbor Problem

3.3.4. Soil Absorption and Saturation Problem

Within oil spill mitigation, the volume of oil or 'liquate' that flows over a given terrain is influenced by several crucial parameters. Among them, the formation of oil puddles, the absorption capabilities of the soil, and the saturation level play significant roles. A profound understanding of these components can substantially enhance the prediction and control of oil spills.

The first parameter, oil puddles, pertains to oil accumulation in certain parts of the terrain. This concept elucidates how oil, rather than distributing uniformly over the surface, tends to gather and form 'puddles' or pools in specific areas, especially in depressions or low-lying regions. Understanding the formation and behavior of these oil puddles is crucial as it affects the volume of oil that spreads during a spill. Calculating the volume of these puddles essentially involves determining the depth and surface area of these pools, which can then be combined to derive the volume of oil contained within.

The next parameter revolves around the absorption or retention capacity of the soil. This characteristic can differ substantially based on the soil type or lithology. Different soils have varying capacities to absorb or hold oil, directly affecting the volume and direction of the oil flow. Recognizing this, leveraging lithology - the study of general rock physical characteristics - can produce more accurate predictions of oil flow paths.

Saturation plays a pivotal role in influencing the behavior of oil spills. Notably, an inverse relationship exists between saturation and absorption. When the saturation level of a terrain rises, its absorption capacity diminishes. This phenomenon is because a highly saturated surface has reached its holding limit, leaving minimal room for further absorption. By adjusting the absorption rate, control over saturation can be achieved. For example, heightening the absorption value decreases saturation, while reducing it leads to increased saturation. This

balance between absorption and saturation is a key determinant in managing the behavior of oil spills on a terrain.

In the computational model tailored for oil spill analysis, two options are provided to account for soil absorption. The first option is useful if detailed soil type maps are available. Users can then employ specific water holding values from experimental studies or prior academic research, reflecting the accurate absorption capacities of various soils, enhancing the model's precision.

The second option, more streamlined, is utilized when detailed soil type information isn't available. Here, a constant value, representing a standard soil water holding capacity, is defined. This value applies uniformly across the model, offering a generalized estimate of the soil's absorption ability. While not as exact as soil-specific data, this approach still furnishes a practical approximation for oil spill prediction and management.

3.3.5. Algorithmic Oil Path Results

Calculating the leakage volume and tracing the path of leaks in oil pipelines are critical for sustainable environmental policies. Another significant aspect of this research is conducting these calculations prior to the emergence of potential issues, thereby enabling the implementation of preventative and protective measures.

The algorithm for calculating leakage volume in oil pipelines proves extremely valuable for pipeline operators in deciding the location of valves, as it operates throughout all points of the given pipeline. Furthermore, once the valve locations are marked on the same algorithm, rerunning the algorithm will allow observation of the potential impact of the proposed valve.

The path calculation algorithm can be applied to all points on the oil pipeline (with submillimeter horizontal resolution) by incorporating a loop into the existing algorithm. However, it should not be overlooked that this process will take a certain amount of time, depending on the length of the oil pipeline and the processing power of the computer being used. The determined leakage path can be cross-analyzed with water sources around the oil pipeline on a GIS basis, which will be useful in deciding where and how to take precautions against potential environmental disaster scenarios.

Figure 9 illustrates a scenario where lithology data is not available. It visually represents elevation values obtained from DEM data, denoted by black numbers. The flow path cells within the blue boundaries indicate the water's path. Within the flow path, the top number indicates the sequential order of the flow direction, ranging from 1 to 23 cells. Red numbers also represent the accumulation amount at the bottom of the flow path. This depiction allows for a clear understanding of the flow direction and the corresponding accumulation values associated with each step along the path.

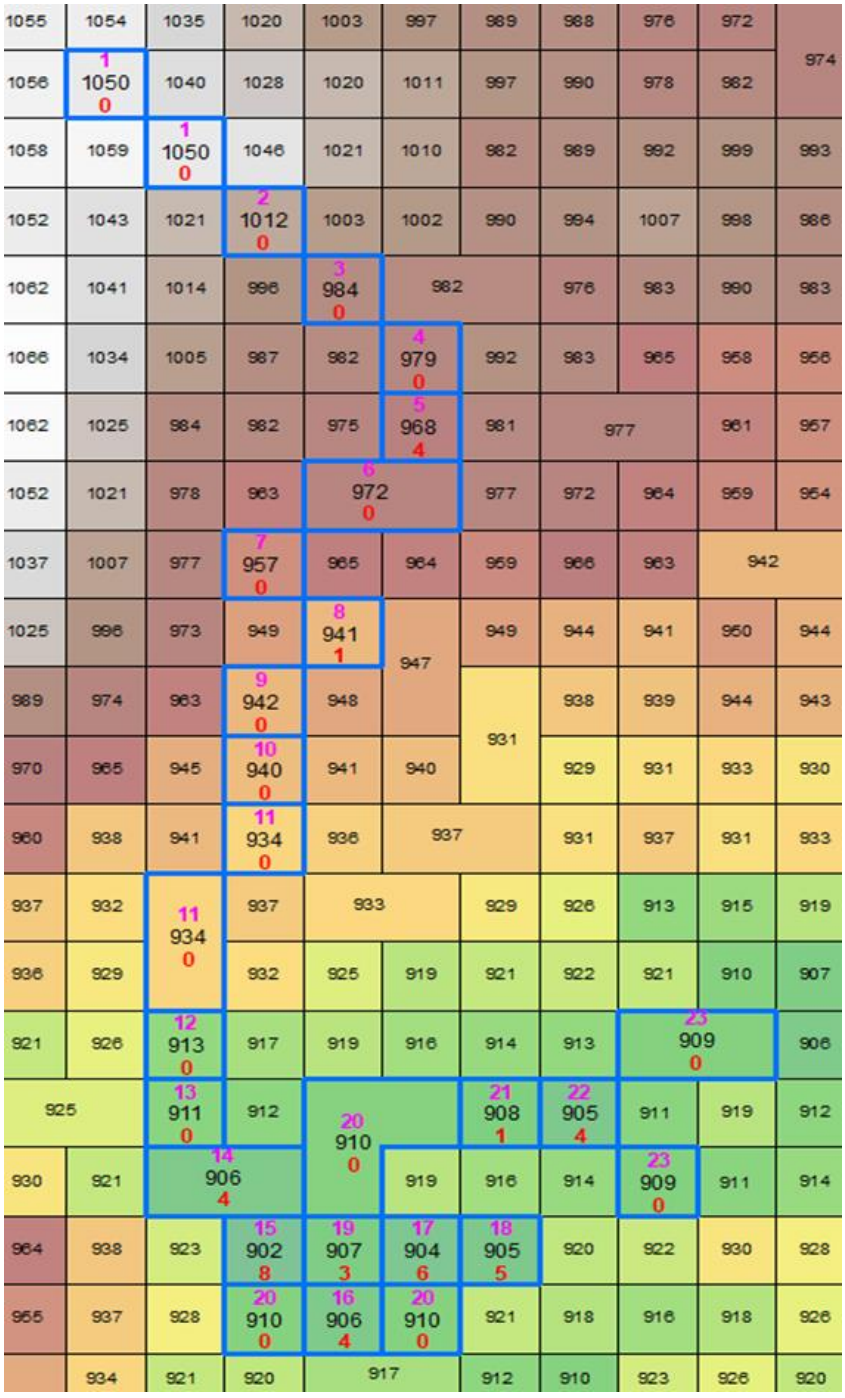


Figure 9. Oil Path Results without Lithology Data

Figure 10 demonstrates the flow path direction when soil type data is available. To incorporate the soil information, the pixels in the image are divided into smaller segments. Each segment represents a different type of soil, characterized by varying absorption rates. The flow path pixels are highlighted in different colors, with each color corresponding to a specific soil type. This color-coded representation aids in visualizing the flow path and identifying the distinct soil types involved. Additionally, the red numbers below the flow path indicate the

sequential order of progress for the individual cells. By incorporating the soil type data, this image provides valuable insights into the flow direction and the influence of different soil characteristics on the overall hydrological process.

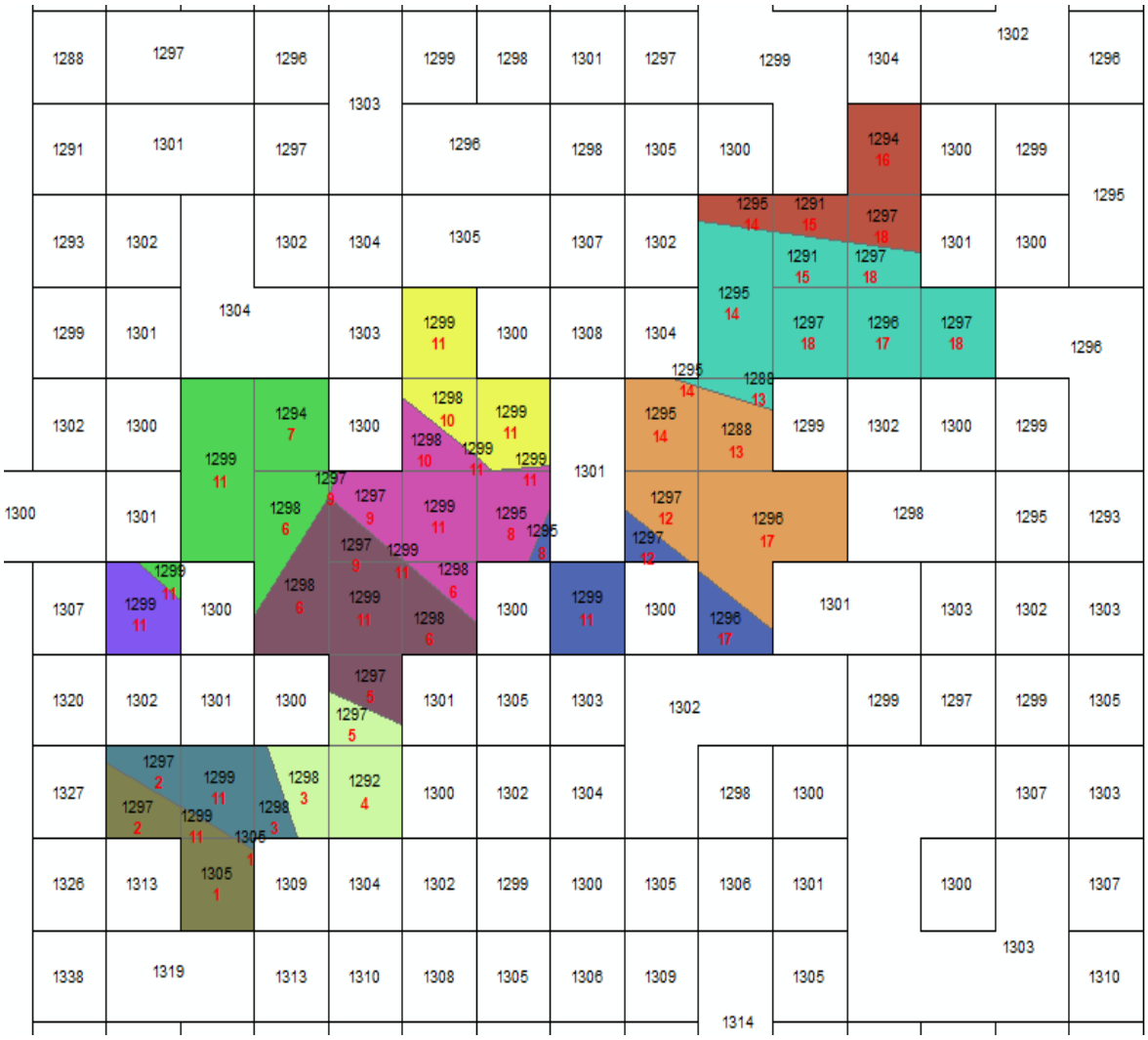


Figure 10. Oil Path Results with Lithology Data

Calculating the leakage path when developing an algorithm for oil pipeline leaks involves initially identifying a single direction from the starting point of the leak. However, considering the possibility of dispersion at the starting point, it would be beneficial to run the algorithm at multiple randomly selected points in areas where a leakage risk is anticipated. This approach would enhance the accuracy and effectiveness of the algorithm in identifying potential leak paths. In Figure 11, the algorithm is executed at randomly selected points, and the results are displayed with a color gradient representing the water accumulation levels,

ranging from green to red. This color-coding effectively illustrates the varying degrees of water accumulation at different points along the pipeline, providing a clear visual representation of the potential impact areas in the event of a leak

Figure 11. Multiple oil spill directions

4. EXPERIMENTAL STUDY

In the "Oil Spill Modelling and Volume Calculation" section, an experimental study has been conducted using photogrammetric methods to test the algorithm described.

4.1. What is Photogrammetry Briefly

Photogrammetry is a complex scientific method that uses sources of electromagnetic radiation, especially photographs, to determine the positions and measurements of objects or areas. Its mathematical foundation is based on geometric and projection equations to process the information from the photographs [11].

This method is widely used in remote sensing, mapping, geographic information systems, urban planning, and many other fields. It processes data from photographs to create three-dimensional (3D) models, maps, or other visual outputs based on mathematical and geometric principles [11].

The key to this process is understanding and correctly applying the photographs' internal and external orientation parameters. Internal parameters provide information about how the photo was taken, defining camera calibration, lens focal length, optical center position, and camera sensor characteristics. This information provides insights about the camera's configuration at the time of the shot [12].

External parameters provide information about where and in which direction the photo was taken, giving details about the photo's geographical location, altitude, and orientation. These parameters are crucial in determining the photo's position and orientation in the real world.

Other essential elements of photogrammetric modeling are tie points and control points. Tie points mark locations that correspond to the same object in two or more photographs and help to relate the images. Control points are spots with known coordinates in the real world and are identified in the photographs. They ensure the accuracy of the model and align the created 3D model with real-world coordinates. Along with these points, stereo image overlay is also vital. It is the process of overlaying two photographs to get 3D depth information. This overlay is possible when two photos overlap by a specific amount. Horizontally, the overlap is not less than typically 60%, and vertically, it is not less than 30% [13]. These overlaps are necessary for accurate 3D data and help in correctly matching the tie points.

In addition to these aspects, collinearity equations play a pivotal role in photogrammetry. These equations describe the mathematical relationship between the 3D coordinates of a point in the physical world and its 2D representation in an image. Essentially, collinearity equations ensure that the lines connecting 3D points to their 2D counterparts and the camera's perspective center are co-linear. This implies that if you extend a line from the camera's lens through a point in the image, it will intersect with the actual location of that point in the physical world. This fundamental principle is crucial for accurately transforming and correlating the 3D and 2D spaces, enabling precise measurements and modeling in photogrammetric applications. By rigorously applying these equations, photogrammetry can achieve high levels of accuracy in mapping and modeling various environments, whether it's for topographical mapping, architectural studies, or archaeological documentation.

$$x_a = x_p - c\frac{r_{11}(\chi_A - \chi_0) + r_{21}(Y_A - Y_0) + r_{31}(Z_A - Z_0)}{r_{13}(\chi_A - \chi_0) + r_{23}(Y_A - Y_0) + r_{33}(Z_A - Z_0)} + dist_x \quad (1)$$

$$y_a = y_p - c\frac{r_{12}(\chi_A - \chi_0) + r_{22}(Y_A - Y_0) + r_{32}(Z_A - Z_0)}{r_{13}(\chi_A - \chi_0) + r_{23}(Y_A - Y_0) + r_{33}(Z_A - Z_0)} + dist_y \quad (2)$$

In the collinearity equations formula (1) and (2);

x_a and y_a stands for image coordinates

X_A, Y_A and Z_A describes for ground coorinates

X_0, Y_0 and Z_0 are external orientation parameters

x_0, y_0 and c defines internal orientation parameters

$dist_x$ and $dist_y$ are for coefficients

Recently, photogrammetry has been used for aerial photos, satellite images, drone shots, and even smartphone cameras. Advances in technology, new algorithms, and software have made photogrammetry faster, more accurate, and accessible. Moreover, it's used in processing high-resolution, multi-band images, surface modeling, vegetation analyses, and even restoration of historical structures. All in all, photogrammetry is an indispensable method to extract metric information about the Earth from photographs.

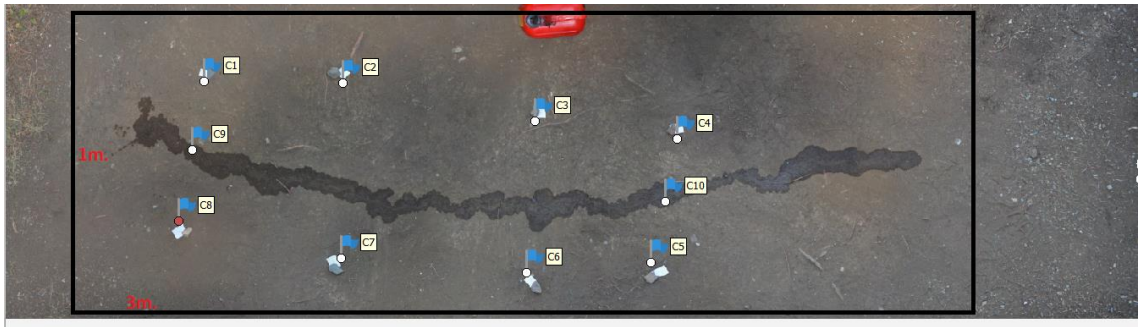
4.2. Study Site and 3D Photogrammetric Models

The experimental study was conducted at the Çankaya location in Ankara. The objective of the experimental study was to recreate a miniature simulation of a real petroleum leak and to evaluate the outputs of a designed algorithm within this context. For this purpose, artificial barriers on the project site have been cleared, making the area ready for application. On the site, 10 control points have been marked with the help of GPS to cover the route of the experimental oil spill. Initially, around 70 high resolution photos were taken with a Canon D5700 to create a DEM (Digital Elevation Model) of the empty surface. After the flow was complete, another set of 70 photos was taken with the same camera for creating orthophoto.

The study utilized 2 liters of gasoline with the aim of accurately simulating a real petroleum leak. The designated area for the experiment was approximately 1 meter in width and 3 meters in length. Control points' positions were measured in

10-minute intervals using GNSS receivers that can connect to the CORS (Continuously Operating Reference Stations) network. When these benchmark points were balanced, the observed total error was found to be less than 7mm. In photogrammetric applications, marking the control points in the captured images resulted in a pixel-based total error observed to be less than 0.5 pixels. Figure 12 displays the control points on the orthomosaic created after the completion of the leak.

For each of the two applications 2 photogrammetric models (before and after leak) were generated using Agisoft Metashape. Figures 13 and 14 display the positions of the captured images.



Reference

Markers	Easting (m)	Northing (m)	Altitude (m)	Accuracy (m)	Error (m)	Projections	Error (pix)
C1	479044.636000	4416462.546000	971.377000	**0.005000**	0.004002	21	0.534
C2	479045.100000	4416462.539000	971.320000	**0.005000**	0.003257	36	0.526
C3	479045.742000	4416462.412000	971.204000	**0.005000**	0.005307	48	0.507
C4	479046.221000	4416462.350000	971.139000	**0.005000**	0.003991	37	0.534
C5	479046.132000	4416461.935000	971.110000	**0.005000**	0.005687	40	0.501
C6	479045.715000	4416461.903000	971.177000	**0.005000**	0.002206	40	0.454
C7	479045.093000	4416461.952000	971.256000	**0.005000**	0.007304	31	0.468
C8	479044.551000	4416462.077000	971.343000	**0.005000**	0.002865	16	0.500
C9	479044.594000	4416462.312000	971.362000	**0.005000**	0.005073	19	0.429
C10	479046.177000	4416462.143000	971.132000	**0.005000**	0.013359	41	0.472
Total Error							
Control points					0.006113		0.495

Figure 12. Error values calculation for the model after the leak

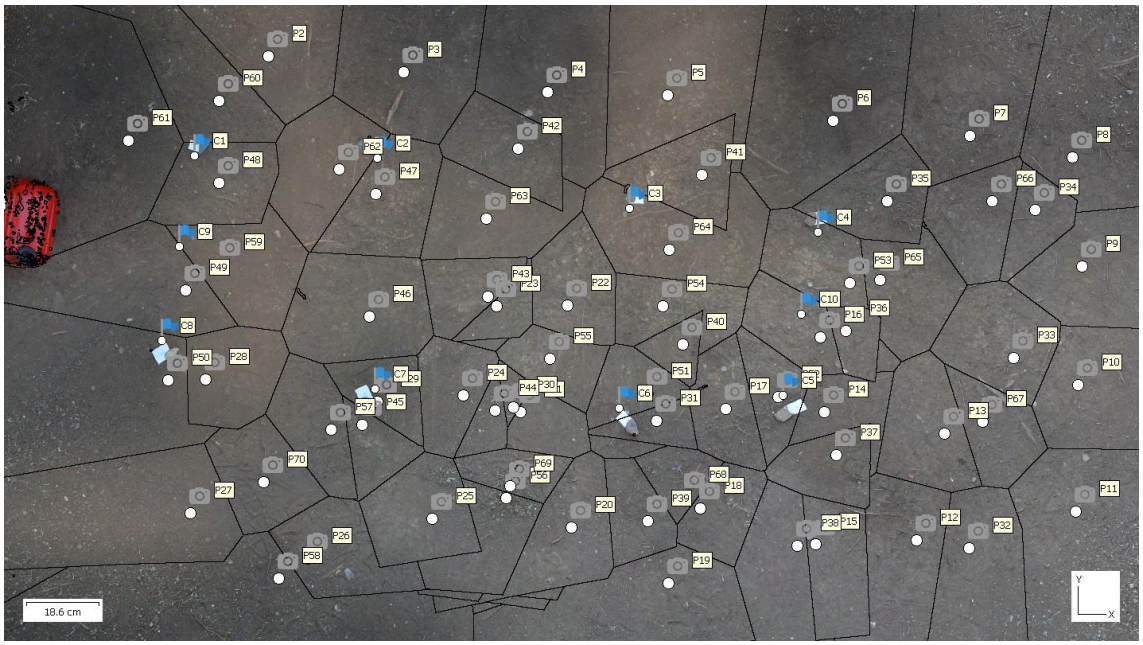


Figure 13. Image and Photogrammetric Benchmarks Locations Before Spill

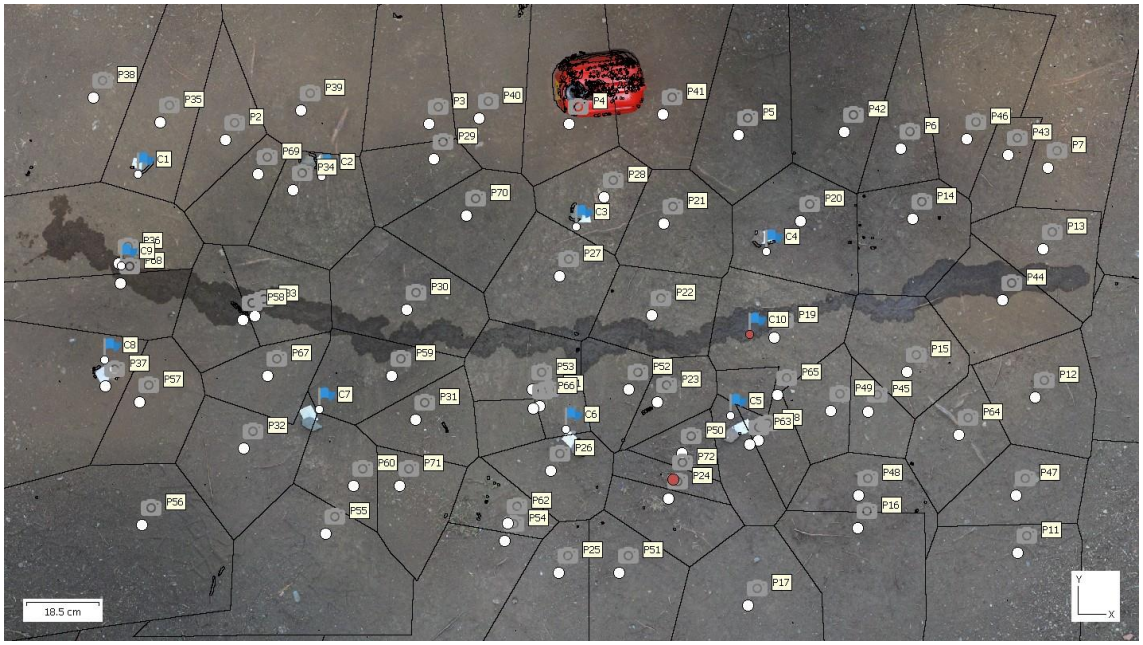


Figure 14. Image and Photogrammetric Benchmarks Locations After Spill

4.3. Camera Calibration and Geneartaing 3D Modeling

Researchers at Hacettepe University have decided to employ the Agisoft application for orthophoto and DEM production due to its convenience in both licensing and 3D modeling. After capturing images, a Nikon D5200 with an 18mm

focal length, camera calibration was performed following the guidelines provided in Agisoft's documentation. A wide-screen monitor was utilized to display a marked chessboard pattern consisting of black and white squares, referred to as a calibration target. At least 10 images of the calibration target were taken from different angles.

By adhering to the given instructions, the necessary parameters for camera calibration were calculated. These calibrated parameters will be used for further processing in the experimental study.

```
<?xml version="1.0" encoding="UTF-8"?>
<calibration>
  <projection>frame</projection>
  <width>6000</width>
  <height>4000</height>
  <f>4607.5382883548918</f>
  <cx>32.819561268028032</cx>
  <cy>-57.553519603999511</cy>
  <b1>0.21571805957467546</b1>
  <b2>1.2697168315769469</b2>
  <k1>-0.10760831260001989</k1>
  <k2>0.043125633396674516</k2>
  <k3>-0.040769766940518273</k3>
  <k4>0.03668132188938427</k4>
  <p1>0.00010069941101074302</p1>
  <p2>-0.00064595178360849035</p2>
  <date>2023-07-16T05:40:52Z</date>
</calibration>
```

Table 1. Camera Calibration Paramters

4.4. Evaluating Oil Spill Route Using With Developed Algorithm

The present study involved the development of an algorithm utilizing the ArcGIS platform to detect pathways of petrol leakage. The analysis and interpretation of outcomes relied upon the DEM data and orthomosaic image generated via the Agisoft software. To accommodate the algorithm's requirement of a 16-bit pixel depth within ArcGIS, the images were rescaled to match this pixel depth.

Furthermore, the horizontal resolution of the DEM produced in Agisoft was initially observed to be around 1mm, which was deemed excessively high for achieving the algorithm's optimal performance. Furthermore, the horizontal length of the petroleum leakage in the narrowest section on the orthomosaic has been measured about 2cm and Considering that the error value of the control points was approximately 0.7 millimeters, it was decided to use a horizontal resolution of 1cm. The vertical resolution of the generated DEM data was scaled to 1mm for the study, as a height change of 0.5 centimeters was calculated in the studied area

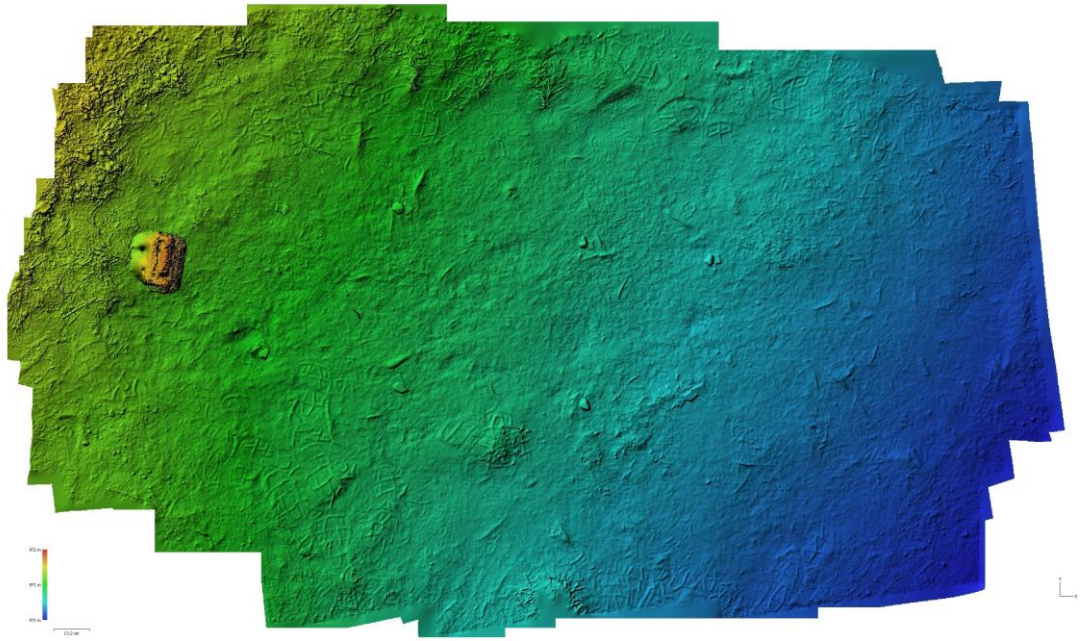


Figure 15. Agisoft Dem Result

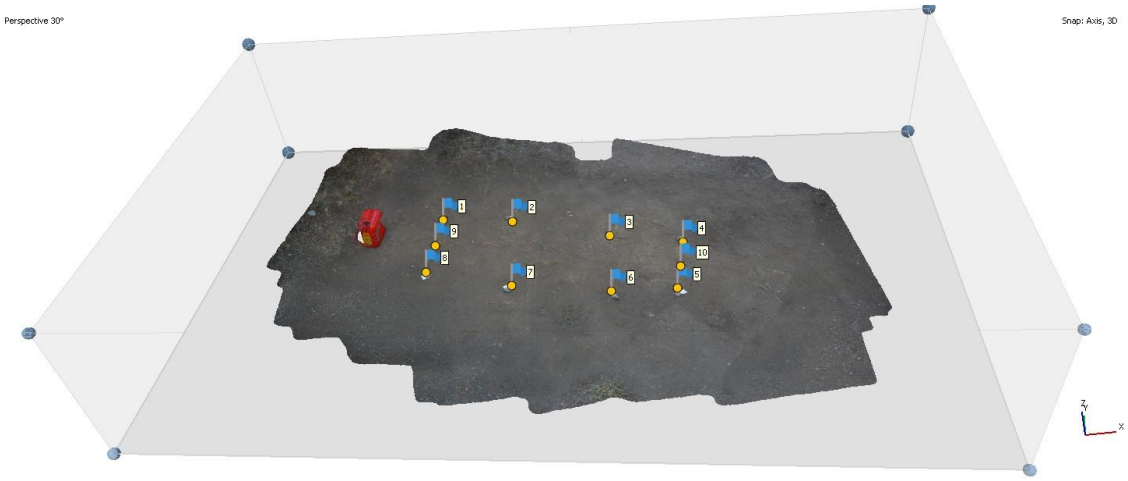


Figure 16. Orthomosaic image of the terrain before the gasoline was spilled

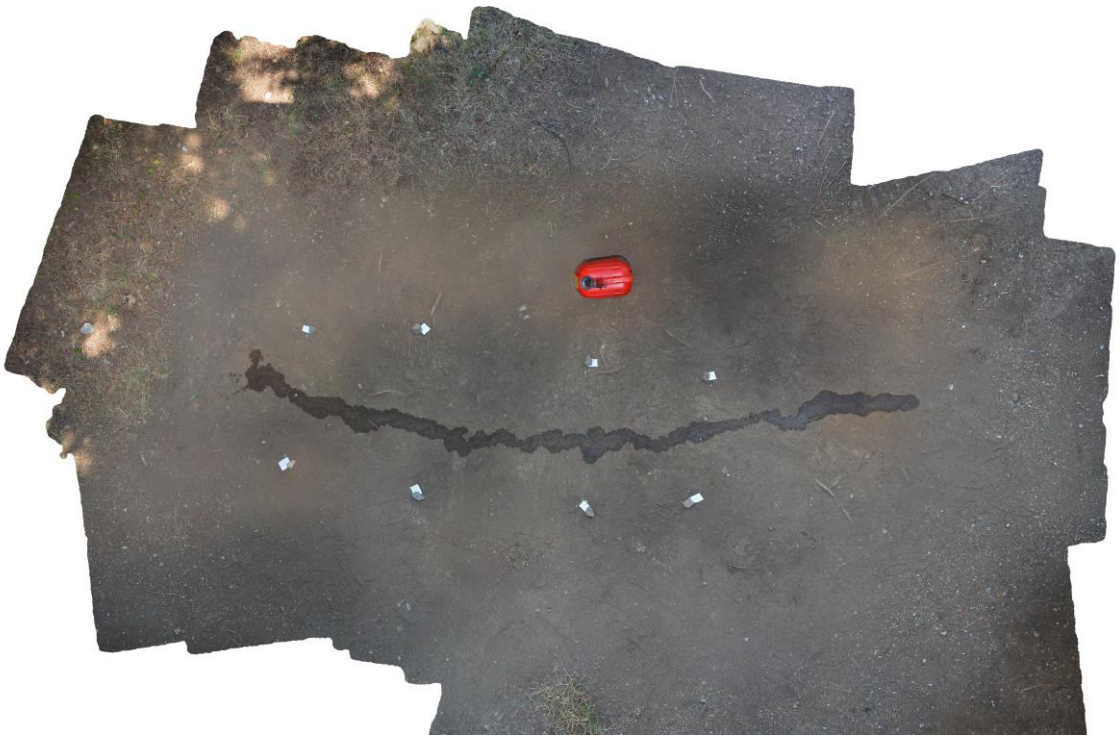


Figure 17. Orthomosaic image after the gasoline was spilled.

Figure 18. Generated Dem and Orthomosaic in ArcGIS

4.5. Experimental Study Results

The DEM and orthomosaic image generated using Agisoft were opened in the ArcGIS program (Figure 15). After the flow was completed, the leakage pathway was digitized using polygon geometry on the produced orthomosaic. Subsequently, the developed algorithm was applied to the re-evaluated DEM data to calculate the algorithmic flow pathway. The actual leakage pathway obtained from the orthomosaic was then compared with the leakage pathway generated by the algorithm (Figure 17).

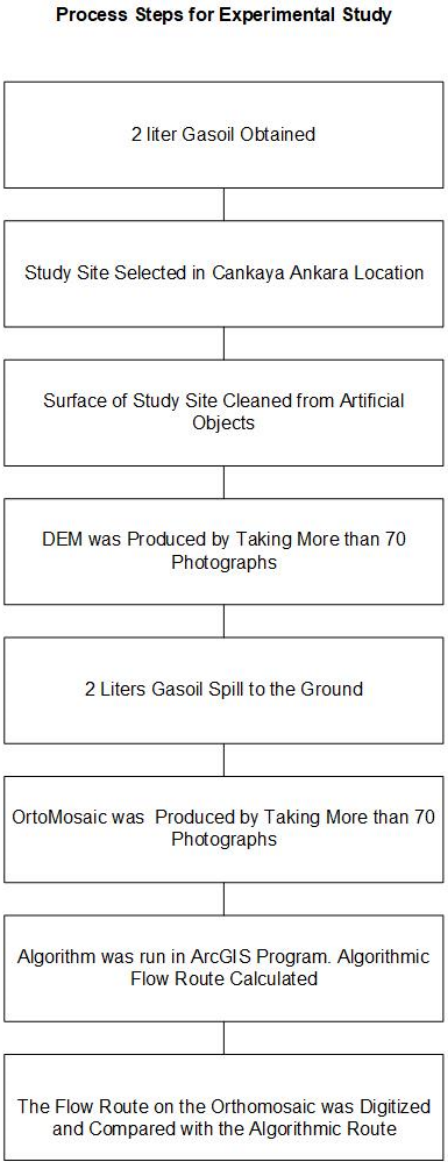


Figure 19. Experimental Study Steps

When comparing the results obtained through the algorithmic calculations with the real-world petroleum leakage route, the following conclusions have been drawn; the algorithm was able to achieve an accuracy of 84.75% when predicting the 2.5-meter-long real-world petroleum leakage route by analyzing 295 pixels out of which 250 pixels intersect with the actual route (Figure 20-21).

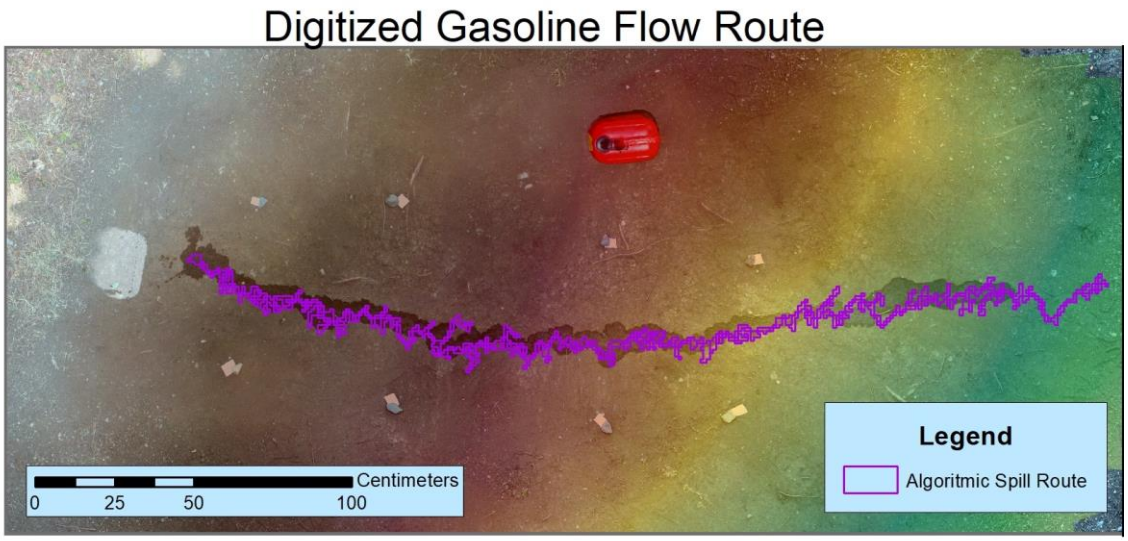


Figure 20. Compresion of Algorithmic and Real Spill Path

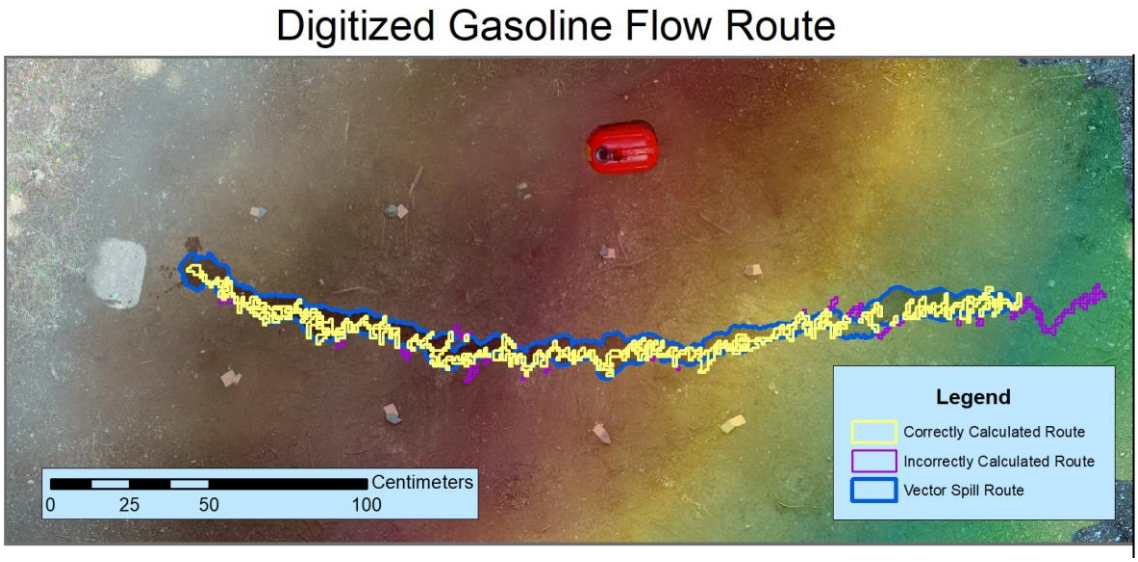


Figure 21. Results of Algorithmic and Real Spill Path

When the reason for the 84% error coming from was examined, it was seen that this problem was caused by the "Neighbor's Neighbor Problem" explained in section 3.3.3. While the algorithm determines pixels of equal height along the flow route, pixels with the same height are also included in the process due to vertical resolution. It was determined that when these pixels were excluded from the

calculation, the pixel accuracy of the model in determining the route increased to 93%. During the process, pixels that were outside the route but provided the connection were included in the calculation (Figure 22-23)

	Accuracy Before Excluding Erroneous Pixels	Accuracy After Excluding Erroneous Pixels
Accuracy	%84	%93

Table 2. Final Oil Spill Result Table

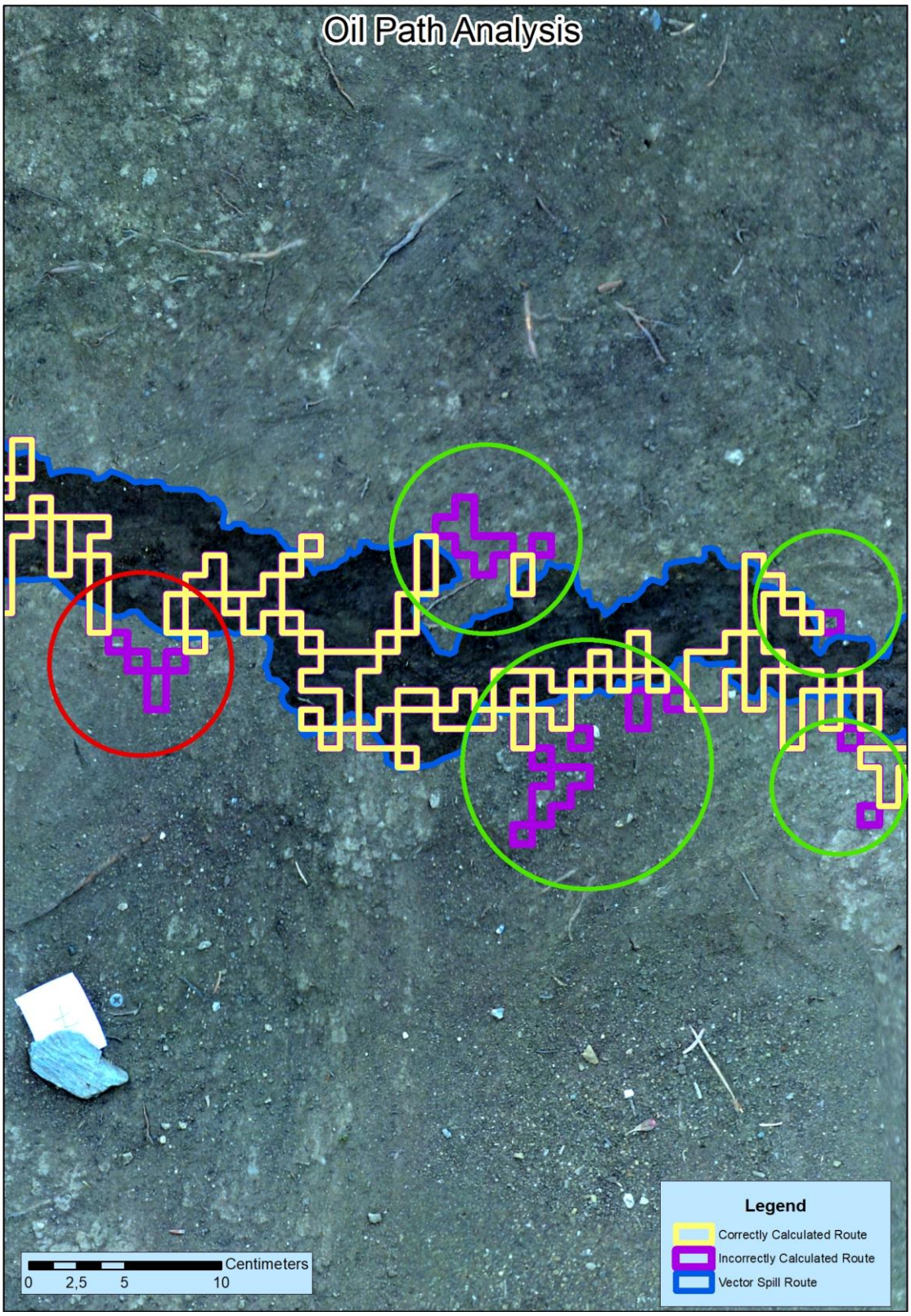


Figure 22. Oil Path Analysis Map

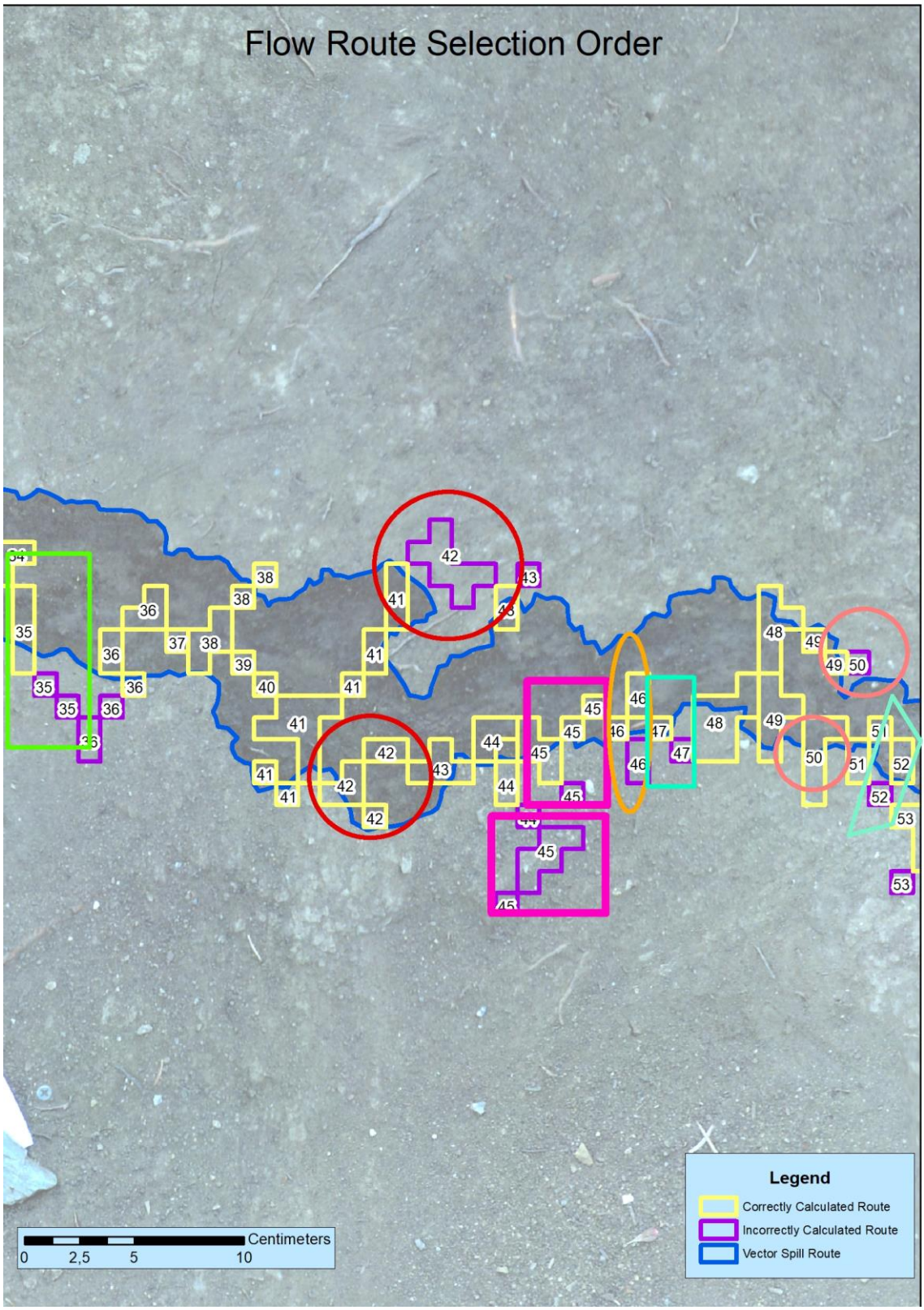


Figure 23. Flow Route Selection Order

5. CONCLUSION

In conclusion, this study has demonstrated the immense potential of Geographical Information Systems (GIS) base programming in planning oil pipeline routes and predicting potential oil spills. The research has first revealed how to find the optimal pipeline route that minimizes topological and geographical complexities. This approach ensures both environmental preservation and socio-economic sustainability. Moreover, the study examines how a determined volume of oil would spread on land surfaces, which will enable the implementation of proactive measures to reduce environmental oil pollution. In both solutions, unique algorithms were developed to enhance route optimization and spill route prediction.

This study has highlighted the importance of topographic classifications during pipeline route selection. These classifications involve identifying distinctive features such as ridges, flatlands, steep terrains, and water channels within a specific area, utilizing advanced GIS analyses. Also, the study has sought to understand the landscape as a continuous unit rather than seeing these components as discrete parts. In addition, the study has also introduced a simple pipeline path by removing complicated patterns and avoiding needless extra turns by using a line simplification algorithm. This strategy results in a more practical and realistic approach toward pipeline construction by removing high vertex points from the proposed route.

Reflecting on the study's results, significant enhancements in pipeline route planning have been observed after the line simplification algorithm was applied to the route. These enhancements include reducing the pipeline's total length from 155.83 kilometres to a more efficient 148.99 kilometres, representing a decrease in algorithmic cost of approximately 20%. This optimization has notably improved the pipeline's interactions with environmental obstacles and barriers.

Furthermore, the findings demonstrate that the results based on topographic classification have noticeably improved.

Furthermore, the research has exploited GIS technology, particularly Digital Elevation Models (DEMs), and state-of-the-art algorithms to predict possible pipeline leakages and spills. High-risk regions can be identified by assessing terrain slope, pipeline pressure, and soil type. This proactive approach enables effective emergency planning and better resource allocation and potentially mitigates the environmental impact of oil spills. In addition to the above, the study suggests that incorporating broader data sets, such as geological and lithological data, could further enhance the accuracy of pipeline route planning and spill prediction. The study also proposes intersecting the pipeline route with watershed boundaries and evaluating the results. This approach could lead to the identification of optimal valve locations, thereby enhancing the efficiency of the pipeline system and reducing the risk of oil spills. However, a limitation of this approach is that initially, the algorithm can identify only a single direction from the leak's starting point. To address this, it is proposed to run the algorithm at multiple randomly selected points in areas with anticipated leakage risks, thereby improving the accuracy and effectiveness in identifying potential leak paths.

The results of the algorithm developed for predicting oil spills have been simulated with real gasoile using photogrammetric acquisition. It has been observed that the spill route calculated by the algorithm aligns satisfactorily with the actual spill route. While the spill route was calculated with an 84% accuracy, it has been found that when pixels miscalculated due to sensitivity are disregarded; the accuracy increases to 93%.

This research has underscored the importance of merging GIS capabilities and innovative algorithms in pipeline route planning and spill prediction, thus reducing potential environmental damage from oil spills. The findings of this study can be instrumental in enhancing the efficiency of pipeline planning and construction,

facilitating effective emergency planning, and mitigating the environmental impact of oil spills.

Future research should continue to explore and refine these methods, further improving the accuracy and efficiency of pipeline route planning and spill prediction. There are widespread crude oil pipelines in the world and terrestrial oil spill for crude oil can be added to the model. The behavior of crude oil, influenced by its viscosity and interaction with different soil types, presents complexities not covered in this study. This integration would greatly enhance the research and could lead to more inclusive models for pipeline spill predictions.

7. REFERENCES

[1] S.E. Chang, J. Stone, K. Demes, M. Piscitelli, Consequences of oil spills

a review and framework for informing planning, Ecology and Society, 19 (2014).
[2] S.Z. Halim, M. Yu, H. Escobar, N. Quddus, Towards a causal model from pipeline incident data analysis, Process Safety and Environmental Protection, 143 (2020) 348-360.
[3] R.D.C.F.S. Silva, D.G. Almeida, R.D. Rufino, J.M. Luna, V.A. Santos, L.A. Sarubbo, Applications of Biosurfactants in the Petroleum Industry and the Remediation of Oil Spills, International Journal of Molecular Sciences, 15 (2014) 12523-12542.
[4] M. Fahimipirehgalin, E. Trunzer, M. Odenweller, B. Vogel-Heuser, Automatic Visual Leakage Detection and Localization from Pipelines in Chemical Process Plants Using Machine Vision Techniques, Engineering, 7 (2021) 758-776.
[5] T. Spyridopoulos, T. Tryfonas, J. May, Incident Analysis & Digital Forensics in SCADA and Industrial Control Systems, IET Conference Proceedings, Institution of Engineering and Technology, 2013, pp. 6.1-6.1.
[6] G. Geiger, T. Hazel, D. Vogt, Integrated SCADA-based approach for pipeline security and operation, 2010 Record of Conference Papers Industry Applications Society 57th Annual Petroleum and Chemical Industry Conference (PCIC), 2010, pp. 1-8.
[7] J.I. Chang, C.-C. Lin, A study of storage tank accidents, Journal of Loss Prevention in the Process Industries, 19 (2006) 51-59.
[8] H. Ibrahim, Hazard Analysis of Crude Oil Storage Tank Farm, International Journal of ChemTech Research, (2018).
[9] U.S.D.o.t. Interior, Pipeline Oil Spill Volume Estimator Paperback, CreateSpace Independent Publishing Platform2015.
[10] S.K. Jenson, J.O. Domingue, Extracting topographic structure from digital elevation data for geographic information-system analysis, Photogrammetric Engineering and Remote Sensing, 54 (1988) 1593-1600.
[11] N.C. Noya, Á.L. García, F.C. Ramírez, Combining photogrammetry and photographic enhancement techniques for the recording of megalithic art in north-west Iberia, Digital Applications in Archaeology and Cultural Heritage, 2 (2015) 89-101.
[12] T. Luhmann, C. Fraser, H.-G. Maas, Sensor modelling and camera calibration for close-range photogrammetry, ISPRS Journal of Photogrammetry and Remote Sensing, 115 (2016) 37-46.
[13] H.A. Sadeq, Accuracy assessment using different UAV image overlaps, Journal of Unmanned Vehicle Systems, 7 (2019) 175-193.
[14] F. 500, Global 500, Fortune 500, 2020.

ATTACHMENTS

APPENDIX 1 – Programming Leakage Volume from Oil Pipelines

```
CREATE TABLE public.petrol_demo
(
    gid integer NOT NULL DEFAULT
nextval('petrol_demo_gid_seq'::regclass),
    gridcode bigint,
    cap smallint,
    parca smallint,
    uzunluk numeric,
    hacim numeric,
    islem smallint,
    orig_fid integer,
    hesap numeric,
    geom geometry(MultiLineString,4326),
    CONSTRAINT petrol_demo_pkey PRIMARY KEY (gid)
)
```

Table 3. Creating Leakage Volume Table on PostgreSQL

```python
#!/usr/bin/env python
#-*-coding:utf-8-*-

import psycopg2
from datetime import datetime
startTime = datetime.now()

# Database connection setup
baglanti_text = "dbname=A_petrol" + " " + "user=postgres" + " " +
"password=postgres" + " " + "port=5432"
table = 'petrol_demo'
conn = psycopg2.connect(baglanti_text)
db = conn.cursor()

# Query to count the number of rows in the table
sorgu = 'SELECT count(*) FROM [14];'.format(table)
db.execute(sorgu)
conn.commit()
kac_oge_var = (db.fetchall()[0][0])

# Function to find the next row to process
def islmid_bul(conn, db, table):
    # ...
    return islem_id, baslangic, yukseklik, hacim

# Function to update the 'hesap' column of a specific row
def math_hesap(conn, db, table, id, hacim):
    # ...

# Get initial values for processing
islem_id, baslangic, bas_yukseklik, bas_hacim = islmid_bul(conn,
db, table)

# Function to retrieve neighboring row based on conditions
def komsuluk(conn, db, id, table):
    # ...
    return [bulunanid, yukseklik, hacim]

ikinci = 0
kactane = kac_oge_var
math_toplam_hacim = 0
yukseklk_maks = bas_yukseklik

# Loop to process all parts of the pipe stored in the database
while True:
    # Check if the end of the loop is reached and reset variables
    if islem_id == -1:
        islem_id = baslangic
        yukseklik = bas_yukseklik
        yukseklk_maks = bas_yukseklik
        ikinci = ikinci + 1

```

```
    # Processing for the second iteration of the loop
    if ikinci == 2:
        # Assumption that approximately half the flow goes to the
pipe where the opening occurs
        math_toplam_hacim = math_toplam_hacim + (bas_hacim) / 2

        # Small value assigned to prevent errors when there is no
flow in the selected part
        if math_toplam_hacim == 0:
            math_toplam_hacim = 0.001

        # Update the 'hesap' column of the first part with the
calculated volume
        math_hesap(conn, db, table, islem_id, math_toplam_hacim)

        # Break the loop when all parts are processed
        if baslangic == kac_oge_var:
            break

        math_toplam_hacim = 0
        islem_id, baslangic, bas_yukseklik, bas_hacim =
islmid_bul(conn, db, table)
        ikinci = 0
        yukseklik = bas_yukseklik
        yukseklk_maks = bas_yukseklik

    # Find the next unprocessed neighboring part
    sonuc = komsuluk(conn, db, islem_id, table)
    islem_id = sonuc[0]
    yukseklik = sonuc[1]
    hacim = sonuc[2]

    # Update the reference height value if the current part's
height is higher
    if yukseklik >= yukseklk_maks:
        yukseklk_maks = yukseklik
        math_toplam_hacim = math_toplam_hacim + hacim

db.close()

# Print the execution time of the algorithm
print(datetime.now() - startTime)
```

Table 4. Pipeline Spill Value Calculator Code in Python

For the algorithm to operate, it is imperative that the table named "petrol_demo" is initially produced in the PostgreSQL database, using the provided code. The resulting table should then be sectioned using the help of geographic analyses into grid cells and filled as shown below. The code can be executed after the Python database connection settings are configured.

gid: This represents the primary key of the table.

gridcode: This denotes the altitude value of each pipeline segment intersected with the DEM (Digital Elevation Model).

cap: This signifies the pipeline's diameter and is used for volume calculations.

parca indicates the remaining sections between the installed valves on the pipeline. For example, the part from the start to the first valve should be numbered 1, and the section from the first valve to the second should be numbered 2.

uzunluk: This is the length of the pipeline segment intersecting with the DEM.

hacim: This is the volume in cubic meters, calculated using the pipe's diameter and length according to the cylinder volume calculation.

islem: This is used by the developed algorithm and stores whether an operation has been previously performed on a cell. The default value to be entered in the database should be 0.

orig_fid: This is a unique value given to each segment produced by intersecting the DEM with the pipeline axis, but it is not used in the algorithm.

hesap: This is the result produced by the algorithm. It holds the calculation of the total volume that will flow from the related part in both left and right directions.

geom: This is the path's geometry, held as a multiline string in geometry type.

Query Editor Query History

```
SELECT * FROM public.petrol_demo
ORDER BY gid DESC LIMIT 100
```

Data Output Explain Messages Notifications Geometry Viewer

	gid [PK] integer	gridcode bigint	cap smallint	parca smallint	uzunluk numeric	hacim numeric	islem smallint	orig_fid integer	hesap numeric	geom geometry
1	175	594	46	4	4.48814477736	48121.5752893281	1	14482	1523957.15197	0105000020E610000...
2	174	595	46	4	41.3854406036	443731.807836111	1	14480	1278030.46041	0105000020E610000...
3	173	594	46	4	27.8315297296	298407.72072596	1	14477	1649100.22469	0105000020E610000...
4	172	598	46	4	13.5538453648	145323.384725077	1	14476	983502.864129	0105000020E610000...
5	171	595	46	4	22.7742697375	244184.131795592	1	14473	1387829.67369	0105000020E610000...
6	170	596	46	4	19.5462054187	209573.051302039	1	14472	1160951.08214	0105000020E610000...
7	169	599	46	4	42.4756512076	455420.969898408	1	14469	683130.686818	0105000020E610000...
8	168	603	46	4	42.4755795759	455420.201868316	1	14465	227710.100934	0105000020E610000...
9	167	601	46	4	2.72917802989	29262.056497336	1	14464	470051.230117	0105000020E610000...
10	166	598	46	4	31.6684403312	339546.808601357	1	14463	825611.207137	0105000020E610000...
11	165	596	46	4	14.8820445746	159564.244020889	1	14462	1075166.73345	0105000020E610000...
12	164	590	46	4	4.26453910275	45724.0908409331	1	14461	1009419.1816	0105000020E610000...
13	163	599	46	4	15.9631279403	171155.544470688	1	14458	570260.030601	0105000020E610000...
14	162	595	46	4	30.8451300645	330719.333341349	1	14457	821197.469507	0105000020E610000...
15	161	589	46	4	26.5805485106	284994.787342146	1	14454	1174778.62069	0105000020E610000...
16	160	585	46	4	70.7358371973	758424.713149688	1	14453	2454913.08409	0105000020E610000...
17	159	585	46	4	70.7358371973	758424.713149688	1	14453	2454913.08409	0105000020E610000...
18	158	582	46	4	21.7879838543	233609.243341765	1	14451	2950930.06233	0105000020E610000...

Figure 24. The Result of Pipeline Spill Value in Database

When the code is executed with the related gids, the seepage flow caused by gravity and topography will be calculated for each segment on the pipeline.

Upon execution, the algorithm generates an output as described above. When it is necessary to rerun the algorithm, it is essential to reset the 'islem' and 'hesap' columns to zero (Figure 24). This requirement is due to the logic based on the developed algorithm.

APPENDIX 2 – Programming Horizontal Oil Distribution

```
import arcpy
arcpy.CheckOutExtension("Spatial")
arcpy.env.overwriteOutput = 1

# The line where the starting point of the oil leak is taken as a
variable with arcpy
baslangic=arcpy.GetParameterAsText(0)
# The line where the volume of oil subject to distribution is
defined with arcpy
petrol_miktari=float(arcpy.GetParameterAsText(1))
# The rate of rise of oil according to horizontal resolution in
very high horizontal resolution dem data is defined with arpy, it
comes from the interface
hacim_orani=float(arcpy.GetParameterAsText(2))
# The user defines whether to enter the lithology value from the
interface
litoloji_var_yok=arcpy.GetParameterAsText(3)
# If lithology is not defined, the absorption amount based on a
stable pixel
pixel_emme=float(arcpy.GetParameterAsText(4))

mxd = arcpy.mapping.MapDocument("CURRENT")

pathmxd=(mxd.filePath).encode('utf8')
pathmxdlist=pathmxd.split("\\")
pathmxdlist.pop(len(pathmxdlist)-1)
pathmxd="\\".join(pathmxdlist)
workspace=pathmxd
arcpy.env.workspace=workspace

# The starting point entered by the user is saved as shp
arcpy.FeatureClassToFeatureClass_conversion(baslangic,
workspace+r"\data\output", "baslangic.shp")

# From the starting point, the dem value is cut from a region
where the most leakage will occur considering the dem horizontal
resolution and from now on, the transactions will be made on this
dem data
arcpy.Buffer_analysis(baslangic,workspace+r"\data\output\kesme_bu
ffer.shp","10 Kilometers","FULL","ROUND","ALL")
arcpy.gp.ExtractByMask_sa(workspace+r"\toolbox\tr_dem_wgs.tif",
workspace+r"\data\output\kesme_buffer.shp",workspace+r"\data\outp
ut\kesme_raster.tif")
# The dem data in raster format is converted to vector format for
processes such as neighborhood analysis
arcpy.RasterToPolygon_conversion(workspace+r"\data\output\kesme_r
aster.tif", workspace+r"\data\output\calisma_alan.shp",
"NO_SIMPLIFY","VALUE")

# New column structures below are added to the dem data converted
to vector format for calculations
arcpy.AddField_management(workspace+r"\data\output\calisma_alan.s
hp", "durum", "SHORT")
```

```
arcpy.AddField_management(workspace+r"\data\output\calisma_alan.s
hp", "yukselme", "FLOAT")
arcpy.AddField_management(workspace+r"\data\output\calisma_alan.s
hp", "alan", "FLOAT")
arcpy.AddField_management(workspace+r"\data\output\calisma_alan.s
hp", "biriken", "FLOAT")
arcpy.AddField_management(workspace+r"\data\output\calisma_alan.s
hp", "lito_emme", "FLOAT")
arcpy.AddField_management(workspace+r"\data\output\calisma_alan.s
hp", "sira", "SHORT")

# Depending on whether the user shows the lithology value, the
dem converted to vector is divided into parts according to the
lithology layer. If lithology is not shown, the dem value is not
divided
if litoloji_var_yok=="Stabil Deger":

arcpy.MakeFeatureLayer_management(workspace+r"\data\output\calism
a_alan.shp", "dem_lyr")
else:

arcpy.Intersect_analysis([workspace+r"\data\litoloji.shp",workspa
ce+ r"\data\output\calisma_alan.shp"],
workspace+r"\data\output\calisma_alan2.shp", "", "" "")

arcpy.MakeFeatureLayer_management(workspace+r"\data\output\calism
a_alan2.shp", "dem_lyr")

# The area value of each pixel is calculated, this value will be
used in accumulation operations
arcpy.CalculateField_management("dem_lyr",
"alan",'!shape.area@SQUAREMETERS!', "PYTHON_9.3")

# The first pixel where the initial leak at the starting point
will start is found by intersection analysis
arcpy.SelectLayerByLocation_management ("dem_lyr", "INTERSECT",
baslangic,"","NEW_SELECTION")
f1 = "GRIDCODE"
liste_komsular=[]
for row in sorted(arcpy.da.SearchCursor("dem_lyr", [f1])):
      liste_komsular.append(row[0])
# The pixel with the smallest height value among the pixels
touching the starting pixel is selected
min_yukselme=min(liste_komsular)

# A value of 1 is assigned to the selected pixels indicating that
the operation has been performed and the height value of the
starting pixel is written to the elevation
arcpy.SelectLayerByLocation_management ("dem_lyr",
"BOUNDARY_TOUCHES", "","","NEW_SELECTION")
experssion = '\"GRIDCODE\" ='+str(int(min(liste_komsular)))
arcpy.SelectLayerByAttribute_management ("dem_lyr",
"SUBSET_SELECTION",experssion)
arcpy.CalculateField_management("dem_lyr", "durum",'1',
"PYTHON_9.3")
arcpy.CalculateField_management("dem_lyr",
"yukselme",'!GRIDCODE!', "PYTHON_9.3")
```

```
111
112 i=1
113 # The loop enters until the total loss amount is greater than or
114 equal to the entered leak volume
115 while True:
116       # The leak route on which the operation has been performed
117 is selected, the loss volume will be calculated on this selection
118       arcpy.SelectLayerByAttribute_management ("dem_lyr",
119 "NEW_SELECTION",'"durum"=1')
120
121       # Total loss volume due to lithology absorption and
122 accumulation is calculated
123       if litoloji_var_yok=="Stabil Deger":
124             arcpy.Statistics_analysis("dem_lyr",
125 workspace+r"\toolbox\tablo.gdb\sonuc", [["biriken",
126 "SUM"],["biriken","COUNT"]])
127             f1,f2 = "SUM_biriken","COUNT_biriken"
128             for row in
129 arcpy.da.SearchCursor(workspace+r"\toolbox\tablo.gdb\sonuc", [f1,
130 f2]):
131                   pass
132
133             deger=row[0]+pixel_emme*row[1]
134       else:
135             arcpy.Statistics_analysis("dem_lyr",
136 workspace+r"\toolbox\tablo.gdb\sonuc", [["biriken",
137 "SUM"],["lito_emme","SUM"]])
138             f1,f2 = "SUM_biriken","SUM_lito_emme"
139             for row in
140 arcpy.da.SearchCursor(workspace+r"\toolbox\tablo.gdb\sonuc", [f1,
141 f2]):
142                   pass
143             deger=row[0]+row[1]
144
145       # If the total loss is greater than or equal to the leak
146 volume, it exits the loop
147       if deger>=petrol_miktari:
148             break
149
150       # All the leak routes processed so far are found, all
151 pixels neighboring this route are selected
152       arcpy.SelectLayerByLocation_management ("dem_lyr",
153 "BOUNDARY_TOUCHES", "","","NEW_SELECTION")
154       arcpy.SelectLayerByAttribute_management ("dem_lyr",
155 "REMOVE_FROM_SELECTION",'"durum"=1')
156
157       # The smallest pixel value among the pixels neighboring the
158 current leak route is determined
159       f1 = "GRIDCODE"
160       liste_komsular=[]
161       for row in sorted(arcpy.da.SearchCursor("dem_lyr", [f1])):
162             liste_komsular.append(row[0])
163
164       min_yukselme=min(liste_komsular)
165       experssion = '\"GRIDCODE\" ='+str(int(min(liste_komsular)))
166
167
```

```
168         # All pixels with the smallest height value among
169 neighboring pixels are selected, these pixels will be assigned as
170 processed pixels later
171         arcpy.SelectLayerByAttribute_management ("dem_lyr",
172 "SUBSET_SELECTION",experssion)
173 
174         onceki_sayi =
175 int(arcpy.GetCount_management("dem_lyr").getOutput(0))
176         sonraki_sayi=onceki_sayi+1
177 
178         # It is checked whether there is a pixel value equal to or
179 smaller than the detected height value around the pixels with the
180 lowest height among the neighbors
181         while True:
182               if onceki_sayi>=sonraki_sayi:
183                     break
184               onceki_sayi =
185 int(arcpy.GetCount_management("dem_lyr").getOutput(0))
186 
187               arcpy.SelectLayerByLocation_management ("dem_lyr",
188 "BOUNDARY_TOUCHES", "","","NEW_SELECTION")
189               arcpy.SelectLayerByAttribute_management ("dem_lyr",
190 "REMOVE_FROM_SELECTION",'"durum"=1')
191               arcpy.SelectLayerByAttribute_management ("dem_lyr",
192 "SUBSET_SELECTION",experssion)
193 
194               sonraki_sayi =
195 int(arcpy.GetCount_management("dem_lyr").getOutput(0))
196 
197         # All newly determined pixels are added to the leak route
198         arcpy.CalculateField_management("dem_lyr", "durum",'1',
199 "PYTHON_9.3")
200         # A sequence number is given to the selected pixels about
201 which order the operation was performed
202         arcpy.CalculateField_management("dem_lyr", "sira",i,
203 "PYTHON_9.3")
204 
205         # If there are pixels with a height value greater than the
206 newly found minimum neighbor pixel height, there will be
207 accumulation in these pixels, the accumulation volume on the
208 detected route is calculated
209         experssion = '\"durum\"=1 AND
210 \"yukselme\"<='+str(min_yukselme)
211         arcpy.SelectLayerByAttribute_management ("dem_lyr",
212 "NEW_SELECTION",experssion)
213         arcpy.CalculateField_management("dem_lyr",
214 "yukselme",min_yukselme, "PYTHON_9.3")
215         experssion = "(!yukselme!-!GRIDCODE!)*!alan!"
216         arcpy.CalculateField_management("dem_lyr",
217 "biriken",experssion, "PYTHON_9.3")
218 
219         # If the absorption value comes from lithology, the total
220 absorption in the pixels is calculated
221         if litoloji_var_yok!="Stabil Deger":
222               experssion = "!emme!"
223               arcpy.CalculateField_management("dem_lyr",
224 "lito_emme",experssion, "PYTHON_9.3")
```

```
225         i=i+1
226         arcpy.AddMessage(i)
227 # The calculated result information is recorded as soon as the
228 total loss amount is greater than or equal to the leak volume
229 arcpy.AddMessage(deger)
230 arcpy.SelectLayerByAttribute_management ("dem_lyr",
231 "NEW_SELECTION",'"durum"=1')
232 arcpy.FeatureClassToFeatureClass_conversion("dem_lyr",workspace+r
233 "\data\output", "cikti.shp")
234 arcpy.AddField_management(workspace+r"\data\output\cikti.shp",
235 "sonuc", "FLOAT")
236 arcpy.CalculateField_management(workspace+r"\data\output\cikti.sh
237 p", "sonuc",'!yukselme!-!GRIDCODE!', "PYTHON_9.3")
238
239 newlayer =
240 arcpy.mapping.Layer(workspace+r"\data\output\cikti.shp")
241 df = mxd.activeDataFrame
242 arcpy.mapping.AddLayer(df, newlayer, "TOP")
243 mxd.save()
244
```

Table 5. Horizontal Oil Distribution Code in Python

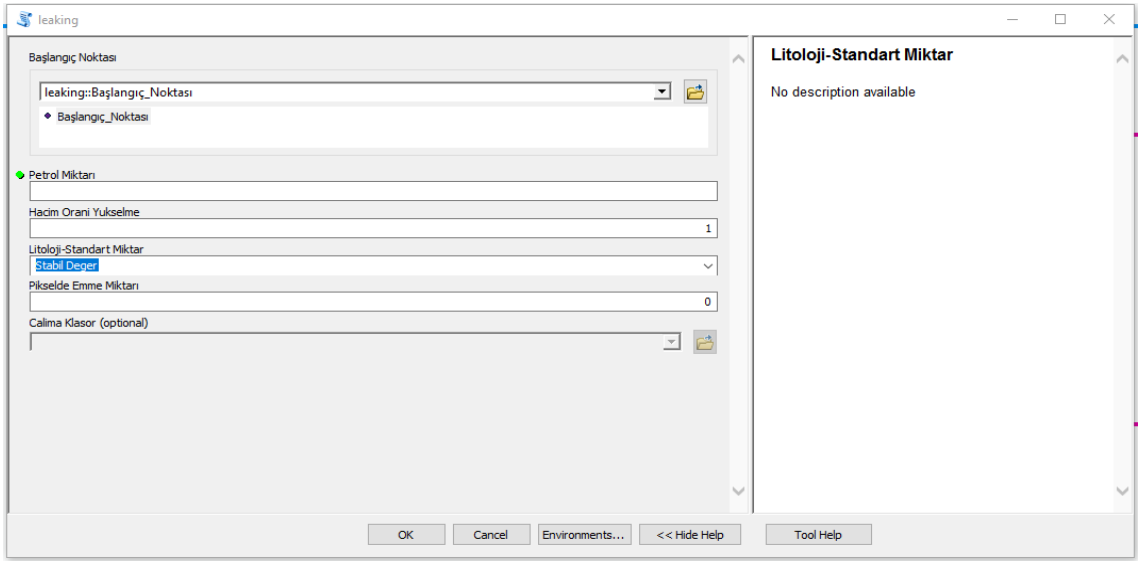


Figure 25. Interface of Oil Spill Distribution Application

Creating an interface within the arctoolbox environment is necessary to properly function the provided code. This interface should include a graphical window displaying a screenshot (Figure 25), allowing users to define input parameters. It should be noted that the DEM and Lithology data are currently hardcoded into the system, without any enhancements to facilitate their modification through the interface.