ETH
Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

RSL Robotic Systems Lab
Prof. Dr. Marco Hutter

Master Thesis

Autonomous Vision-based Safe Proximity Operation of a Future Mars Rotorcraft

Autumn Term 2024

Supervised by:
Roland Brockers
Robert Hewitt
Marco Hutter

Author:
Lasse Fierz

Contents

1 Methodology **2**
- 1.1 Stereo Camera . . . 2
  - 1.1.1 Stereo Camera Advantages . . . 3
  - 1.1.2 Theoretical Analysis . . . 4
- 1.2 Ground Truth Depth . . . 5
  - 1.2.1 Ground Truth Implementation . . . 5
  - 1.2.2 Comparability to SFM . . . 5
- 1.3 Autonomous Landing Procedure . . . 6
- 1.4 Landing Site Handling . . . 7
  - 1.4.1 LSD Properties . . . 7
  - 1.4.2 Landing Site Heuristic . . . 8
  - 1.4.3 Landing Site Manager . . . 11
  - 1.4.4 Implementation . . . 11

2 Stereo Camera Depth Node Implementation **12**
- 2.1 Implementation . . . 12
  - 2.1.1 Stereo Setup Overview . . . 12
  - 2.1.2 Input Handling . . . 13
  - 2.1.3 Disparity Creation . . . 15
  - 2.1.4 Point Cloud Creation . . . 15
  - 2.1.5 Switching . . . 16
  - 2.1.6 Landing Site Detection without Lateral Motion . . . 17
- 2.2 Qualitative Practical Analysis . . . 19

Bibliography **22**

List of Acronyms

- **UAV:** Unmanned Aerial Vehicle
- **SFM:** Structure From Motion
- **LSD:** Landing Site Detection
- **LS:** Landing Site
- **BA:** Bundle Adjustment
- **DEM:** Dense Elevation Map
- **OMG:** Optimal Mixture of Gaussian
- **LOD:** Level Of Detail
- **HiRISE:** High Resolution Imaging Science Experiment
  (High Resolution Satellite Imagery on the Mars Reconnaissance Orbiter (MRO))
- **LRF:** Laser Range Finder
- **GT:** Ground Truth
- **LSM:** Landing Site Manager
- **FSM:** Finite State Machine
- **BT:** Behavior Tree

Chapter 1

Methodology

As the endeavor of this thesis was the merger and enhancement of various aspects of the LORNA project, the complexity lied rather in the understanding of the existing work and the interfaces thereof as opposed to the challenging methodology pursued in the novel contributions. Therefore, the implementation aspect of this work carries more weight than the theoretical decision-making associated with it.
The autonomy framework[1] allows us to fly independent missions at cruise altitude of 100+ m. The structure from motion approach captures 3D information during traversal as its adaptive baseline allows it to perceive high quality depth information also at such high altitudes. This information can be used by LSD in order to detect landing sites during mission.
At low altitudes SFM works as well but surrounded with obstacles, the need for lateral motion poses significant risk. This is because a local state estimator is by nature prone to accumulate an estimation error and the same holds for the structure from motion approach. Our terrain knowledge can thus become void.
To overcome this issue, a range sensor can be used. As LiDAR might come to mind. However, a LiDAR sensor produces rather sparse point clouds unless a newer sensor like for instance Ouster's OS1-128 sensor is used. In that case, there remains the weight issue with the sensor's 495g. As the drone is going to fly on Mar's very thin atmosphere, this isn't feasible.
Staying with the project's theme of visual sensors, a stereo camera poses a solution as it offers in-place triangulation with a very low weight ($\tilde{1}0g$). Therefore, in this thesis I present a stereo camera range node implementation to remedy the shortcomings at low altitudes.

1.1 Stereo Camera

The implementation of the stereo camera sensor itself is very straightforward as simply duplicating an existing camera, offsetting it an adequate distance to resemble the real model, and setting the parameters to equal the hardware, results in the desired outcome.
The input to the stereo camera depth node are the two camera images and the drone's base link pose. Processing this information is different from for the existing SFM algorithm. Therefore, a new depth generation node was put in place.
As mentioned in **??**, state-of-the-art deep learning based stereo depth methods have considerably higher computational overhead. Due to the embedded CPU's computation limitations, this restricts us to using classical algorithms such as OpenCV's implementations of Heiko Hirschmüller's approach ([2]).
The initial goal of the stereo camera implementation of this work was to show proof

of concept for this approach of depth detection without lateral motion. Due to personal experience with the OpenCV library, the initial choice of stereo depth method was OpenCV's StereoSGBM algorithm. This algorithm is introduced in section 2.1.3.
JPL has its own visual library called JPLV containing a stereo matching method which was also considered throughout the time of this thesis. However, as there was no specific reason to switch away from the working StereoSGBM implementation, the continuation thereof was pursued.
Additionally, OpenCV's StereoBM variation was considered which is in general faster but less accurate than StereoSGBM. An analysis thereof is shown in section 2.1.3.

1.1.1 Stereo Camera Advantages

The specific advantage of a stereo camera implementation when compared to SFM can be summarized in the following points:

- No necessity of lateral motion
- Hardware depth perception
- DEM conversion
- Efficiency

Lateral Motion

As already mentioned above the need for lateral motion in itself is an undesirable necessity for a rotorcraft in unknown terrain.
In this setup the structure from motion approach is based on a key frame buffer which needs to be filled with image-pose pairs at different horizontal positions in order to start acquiring depth information. The current setting in the implementation Domnik et al. [3] uses 6 key frames. Therefore, for a single point cloud it is necessary to move laterally 6 times in order to start perceiving depth. Following the depth error formula from a stereo disparity image (**??**) and assuming an altitude of 2.5 m above ground with a focal length of 256 pixels and a disparity error of 0.5 pixels, the necessary baseline in order to keep the depth error below a critical 5 cm is:

Software vs Hardware Depth Perception

Structure from Motion, being a software node that relies on camera poses supplied by a state estimator, is by design subject to inaccuracies. A depth node based on a stereo camera on the other hand works with a fixed rigid baseline between the camera views. Thus, for low altitude flights that bear the danger of collision, a more robust hardware approach is preferred.

DEM Conversion

As described in **??** the multi-resolution DEM used for depth aggregation in LSD is based on Optimal Mixture of Gaussian cells and thus converges over time.
According to **??** the landing sites chosen are likely on terrain with low uncertainty. Because of this landing sites are more likely to be detected and have in general a better quality when the terrain perceived has been viewed.
When a landing site has been selected we need to make sure that the landing site is actually correctly detected and of good quality. For this we would like to (re-)detect

landing sites on rather converged terrain. Structure from Motion needs constant lateral motion for this. A stereo camera depth node simply hovers in place for any given amount of time.

Efficiency

All in all the stereo camera setup allows us to perceive a landing site at course altitude and after having traversed horizontally to that location, we can simply descend to a stereo camera friendly altitude for the verification. Compared to repeated lateral coverage of the area in question this is a huge increase in efficiency. Looking in depth at the stereo alternative of depth generation, we can first analyze the theoretical threshold of this system.

1.1.2 Theoretical Analysis

When it comes to depth perception the obvious drawback of a stereo camera is its limited baseline. It only perceives depth accurately for objects within a certain proximity to the lens.
Assuming a perfectly calibrated and rectified camera there is still always an inaccuracy in the depth estimation arising from the disparity error.
The depth error is estimated using the following derivation. The formula to derive a depth value from a calculated disparity is:

$$z = \frac{f \cdot b}{d} \tag{1.1}$$

Where b is the z is the depth estimate, b is the baseline, f is the focal length and d is the disparity value.
Taking the derivative of z w.r.t. d we get

$$\frac{\partial z}{\partial d} = -\frac{f \cdot b}{z^2} \tag{1.2}$$

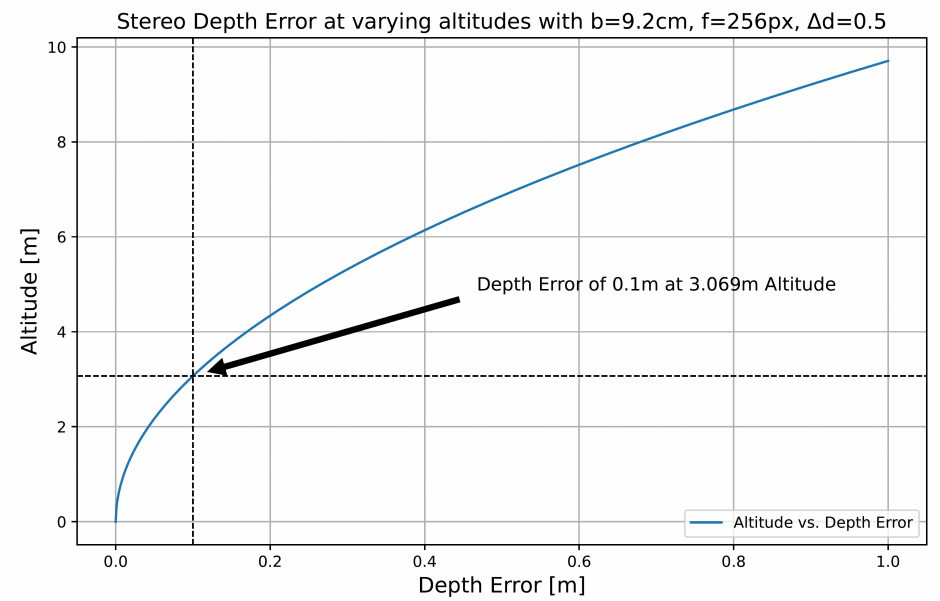
And substituting (eq. (1.1)) we get:

$$\partial z = \frac{z^2}{f \cdot b} \partial d \tag{1.3}$$

Where the sign was left away as for our application there lies equal danger in a point being perceived too close and too far away.
For the maximum altitude given a maximum allowable depth error this yields:

$$z_{\max} = \sqrt{\frac{\Delta z_{\max} \cdot b \cdot f}{\Delta d}} \tag{1.4}$$

Where Δz is the depth error and Δd the disparity error.
The stereo camera mounted on the drone in JPL's aerial vehicle lab had a baseline of about 10 cm and a focal length of 256.
With these properties and estimating a subpixel-precision disparity error of 0.5 pixels the depth error at varying altitudes looks as follows:
Let's assume we allow a maximum depth error of 10 cm. Considering this constraint we can fly at a maximum altitude of about 3 m as indicated in section 1.1.2.
This limitation has to be kept in mind. However, it is neither too surprising nor is it too restrictive as the stereo camera is simply a depth alternative for low altitude flight maneuvers. In the context of an entire science mission it is almost exclusively used for landing site verification purposes.



1.2 Ground Truth Depth

For evaluation purposes as well as proof of concept aspirations, a ground truth is required.
Additionally, GT was important, because at the time of this work the structure from motion node showed frequent signs of unreliability. See **??** for the evaluation thereof.

1.2.1 Ground Truth Implementation

The simulation already supplied the ground truth pose of the drone's base link through the ROS bridges. When applying the static camera transform to it, this yielded the ground truth camera pose. Using Gazebo's depth camera sensor[1], a ground truth point cloud could be created.
The depth camera creates the image using traced rays which fill a pixel with the center most range value that a ray detected.
As expected, the ground truth point clouds yielded very clean and easily interpretable LSD DEMs:

1.2.2 Comparability to SFM

This is sufficient for a qualitative analysis of the stereo depth node. However, to use the ground truth as an alternative for SFM, one has to make sure that the GT quality is not too good. For instance, SFM, like the stereo camera depth, has a depth error associated with its point cloud creation. At high altitudes this can lead to the neglect of small (but for the drone threatening) rocks. So, in order to test the autonomous landing pipeline with ground truth, I had to make sure that small rocks of about 10 cm diameter were not seen by the GT at a cruise altitude of 100 m.
For this, the following test was designed.

[1] As there was a bug in Gazebo's source code, the depth camera couldn't be used out of the box. More on this in **??**.

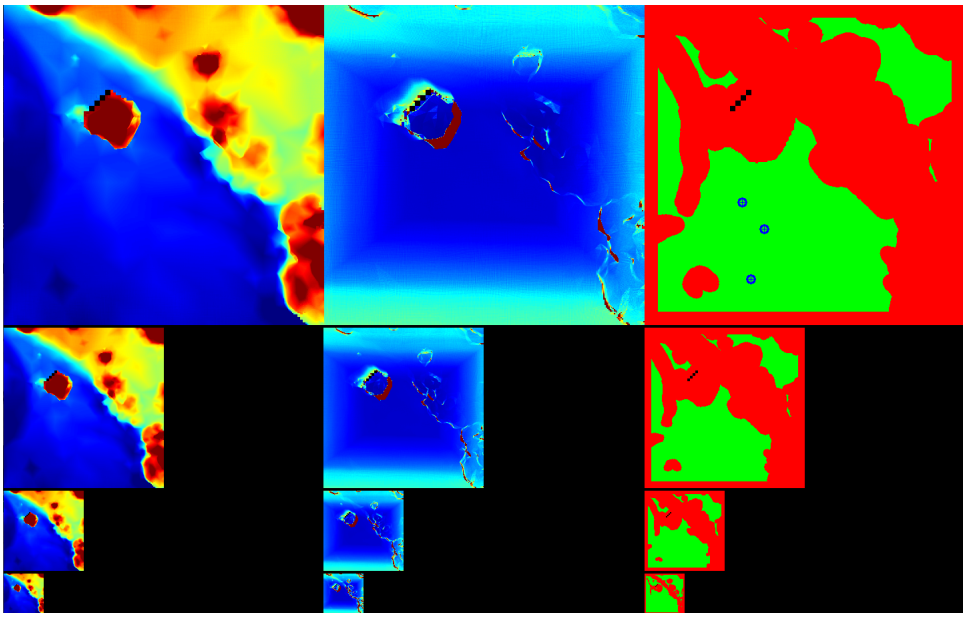


Figure 1.1: LSD debug image using a GT point cloud

On a flat textured plane, three cylinders of different sizes were placed. On each cylinder a small rock of 10 cm diameter was placed. The drone was then flown over the test setup to detect the scene once with SFM and once using the GT depth. The goal was to find out, whether the ground truth depth quality would have to be artificially decreased, using Gaussian or median filtering for instance, in order to make it more comparable to SFM.
Looking at fig. 1.4 and fig. 1.5, one can see, that, though the SFM quality is visibly worse, neither SFM nor GT detected the small rocks on the platforms. This can be seen because in both Debug images the center of the platforms was considered a landing site even though there was the rock which should definitely prevent the detection of a valid landing site.
Therefore, the ground truth depth could be used for the testing of the autonomous landing pipeline without making a landing site verification step at low altitudes redundant.

1.3 Autonomous Landing Procedure

Having implemented a stereo camera as a low altitude alternative to SFM, and after ensuring a correct ground truth comparison, the main contribution of this work could be tackled: Bringing the visual landing site pipeline together with the autonomy framework in order to achieve reliable autonomous landing in unknown terrain.
The approach of the landing pipeline implementation can be split into the following parts:

- Landing site detection output

  Prior to this work, LSD only published one single landing site's location. To give the autonomy more information to make adequate decisions and to avoid the stagnation on a single overconfident landing site, LSD was changed to also first, publish three landing sites each iteration and secondly, yield additional characteristics with each output landing site.

Figure 1.2: LSD debug image simulation terrain reference

- Landing site interface of the autonomy

  The autonomy framework is changed to correctly receive and handle incoming landing sites. The incoming candidates are ordered according to a novel landing site heuristic and updated upon being re-detected.

- Autonomous landing behavior

  Using the newly expressive landing sites and their handling procedure, the adaptive landing instance is put in place using an existing behavior tree framework within the autonomy. Additional modular action nodes are created and previously existing ones are altered in order to achieve a precise and safe landing procedure.

1.4 Landing Site Handling

1.4.1 LSD Properties

With the low altitude depth alternative in place, the connection of the autonomy with the landing site detector could be tackled.
Before this work the output of the landing site detection algorithm was merely the location of a found landing site. However, as described in **??** the landing site detection algorithms segments hazards based on roughness and slope. Subsequently, it considers the size of a landing site as well as the uncertainty associated with a certain selected location.
Simply outputting the location of a landing site is therefore a waste of information when so many characteristics are at hand to make an informed selection.
I decided on the following properties to be the LSD output:

- Location

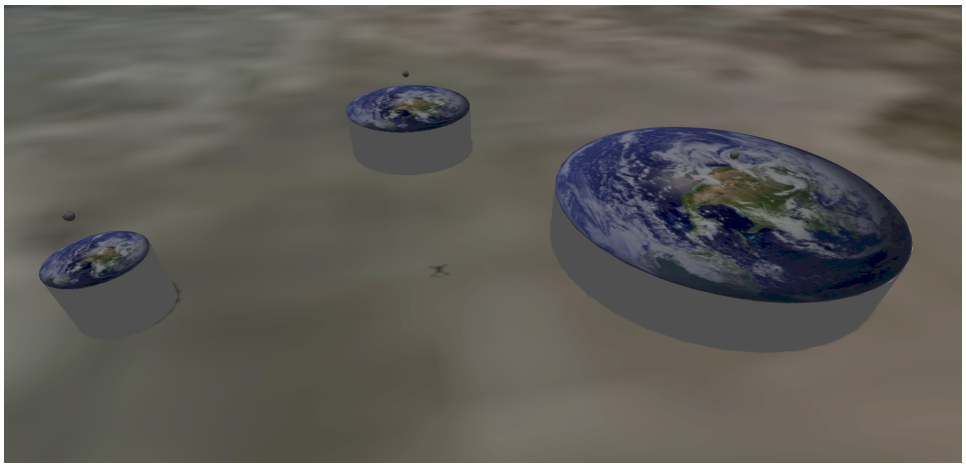


Figure 1.3: GT test setup: Rocks of 10 cm diameter where placed on platforms of different sizes to be detected by both the ground truth and SFM

- Uncertainty
- Roughness
- Size
- Obstacle Altitude

The final landing site detection output is a custom landing site ROS message containing the above-mentioned characteristics of the detected spot.
Location
The location of the landing site in the world frame. **Roughness**
The roughness value the exact value already used for the hazard segmentation step in the landing site detection.
Uncertainty
The uncertainty value is also a product of the landing site detection algorithm. It denotes the averaged uncertainty across the area around a given landing site. The uncertainty of a single map cell denotes the stereo depth error estimates merged over time.
Size
To determine the size of a landing site, the landing site detection algorithm performs a distance transform on the created landing site map in order to find the closest non-landing site for any found landing site. This returns the radius of the largest valid landing circle around a landing site. Calculating the physical value, the metric radius is returned as the size of a landing site.
Obstacle Altitude
The obstacle altitude was newly introduced in this work. It defines the current highest point of the aggregated DEM's highest resolution layer. As no actual object detection is performed and no hazard information is retained in this visual pipeline, this value serves the autonomy as an indication of the obstacles heights to avoid in the vicinity of a certain landing site. More on this in section 1.4.

1.4.2 Landing Site Heuristic

The autonomy processes the in section 1.4.1 listed values in order to arrive at the following final landing site properties:

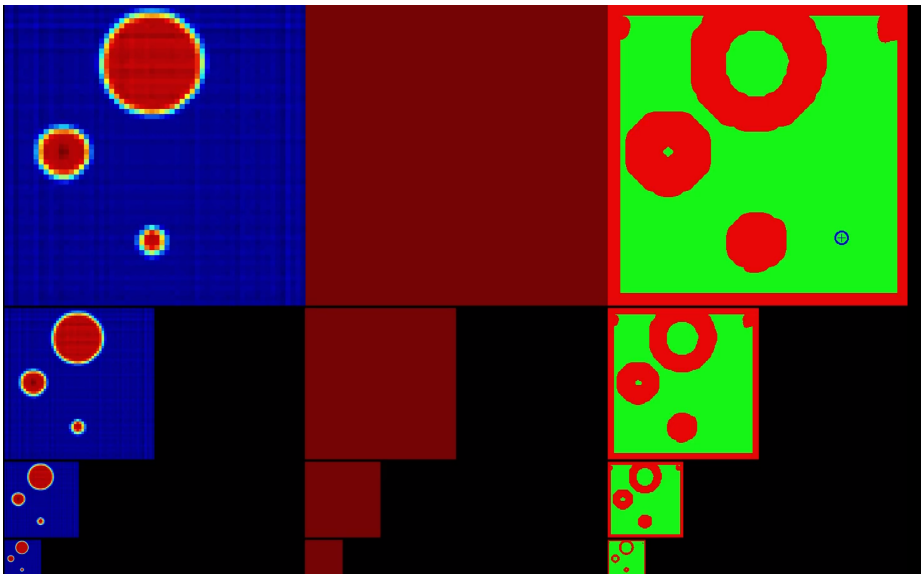


Figure 1.4: Ground truth result of the GT test

- Current Distance to Drone
- Roughness
- Uncertainty
- Size
- Verification Altitude

The final heuristic defining the quality of a landing site is in fact a square loss function:

$$L_{LS} = w_{dist}L_{dist} + w_{rough}L_{rough} + w_{var}L_{var} + w_{size}L_{size} + w_{verAlt}L_{verAlt} \quad (1.5)$$

Current Distance to Drone - L_{dist} Well likely the single most important characteristic of a landing site[1]. Each iteration the current distance to the drone's position is calculated for each retained landing site. The distance is then normalized by dividing it by the cruise altitude which is 100 m. In practice there were easily enough landing sites found while moving to allow landing sites to fall off when being farther away than 100 m.
Roughness - L_{rough}
The roughness property is the unaltered roughness value received from LSD. It is already normalized and enters the loss function as it is.
Uncertainty - L_{var}
The same holds for the uncertainty. It is already normalized by design and enters the loss function unaltered.
Size - L_{size}
Analogous to the roughness and uncertainty properties the size comes from the landing site detection directly. However, unlike the two preceding properties it is not normalized but simply denotes the metric radius of the largest circle of valid

[1]Each received landing site has already undergone a threshold filtering regarding slope and roughness.

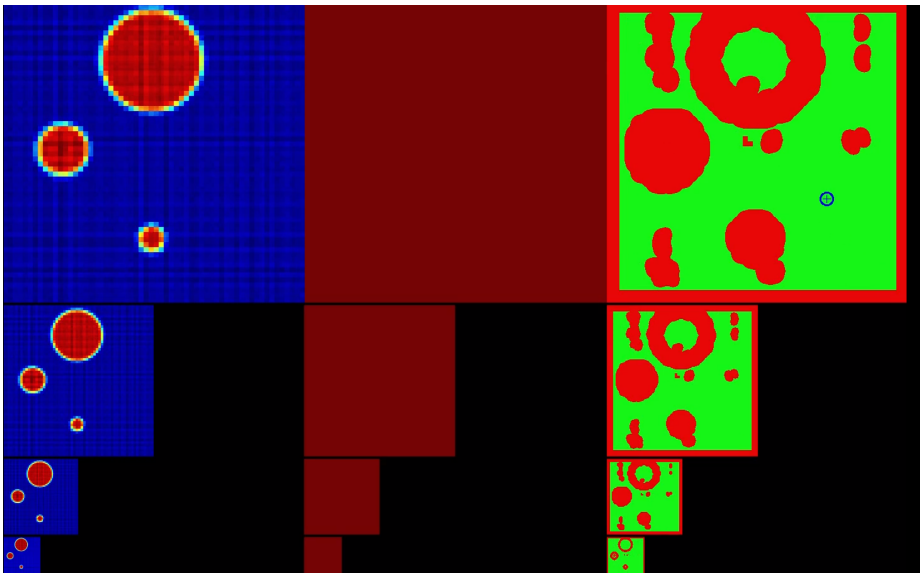


Figure 1.5: SFM result of the GT test

landing area that can be fit around a given landing site. This is achieved in LSD by performing a distance transform on the created landing site image.
In order to normalize this value the maximum landing site size is retained and each landing site's size is divided by it in order to achieve normalized size information. Also, as can be seen in eq. (1.5), the size contribution enters the loss function with a negative sign. This is due to the fact that compared to all other characteristics, the size defines a property that we would like to maximize.
Verification Altitude - L_{verAlt}
A site's verification altitude is the smallest vertical distance between the drone and the landing site at which that site was (re-) detected.
The verification altitude is a useful property because of numerous reasons.

- Further Indication of Certainty

  First, similar to the uncertainty metric the verification altitude indicates how certain we can be about a detected landing site as spots detected at lower flight altitudes are more likely correct due to the reduced depth error. Even though it might seem overlapping with the uncertainty property in this regard, these two characteristics are quite complementary as the uncertainty takes OMG convergence and camera specifics into consideration while the verification altitude is a purely location based metric.

- Landing Site Property Updates

  As the verification altitude yields a simple and good estimation of the trustworthiness of an incoming landing site, it can be used as a flag to know, when a landing site's properties should be updated. When a landing site is re-detected with a verification altitude lower than the previously stored one, the algorithm trusts it more and alters the previously stored properties to the new ones received.

- Verification

  Continuously updating the verification altitude upon re-detection allows us to determine the lowest altitude, at which a landing site was re-detected. This

information can be used to verify that a given site was considered a valid landing spot even at low altitudes.

1.4.3 Landing Site Manager

Prior to this thesis, the landing site manager received artificially generated landing sites from a dummy landing site node. In this work the LSM was connected to the actual landing site detection output.

1.4.4 Implementation

With the interface and the input fixed, the actual implementation could be tackled. This is explained in detail in **??**.

Chapter 2

Stereo Camera Depth Node Implementation

2.1 Implementation

Like Structure from Motion, the stereo depth instance is a ROS node which is given images and image poses from the xVIO state estimator. As the state estimator was in its final development stages during my thesis, camera images and a ground truth camera pose from the simulation were used instead as input for the stereo algorithm. Note that only one camera pose is given as the second one is derived in a straight forward manner, given the fixed hardware baseline.

2.1.1 Stereo Setup Overview

fig. 2.1 shows the drone setup in the simulation with the stereo camera. The stereo camera pair is indicated with the opaque boxes. The significant distance to the drone's core is necessary to avoid capturing the landing feet in the image due to the simulation model's discrepancy to the physical drone. As presented in **??** the drone hardware has landing skids that are spread significantly farther apart than for the simulated model. That is why, when using the stereo camera mounted at the core of the physical drone, the mainstays are not visible in the images detected. In the simulation they would be detected unless the stereo cameras are positioned further from the rotorcraft's center.

Frames

A critical part of navigation is always the consistency of the coordinate systems in which quantities are represented. Hereafter in fig. 2.2 the present coordinate systems of the stereo camera setup are displayed.
Notably there are three important frames:

- The reference world frame W

  This is the global frame relative to which the drone flies and the point clouds are created.

- The drone (base link frame) D

  This is the pose of the moving drone throughout a mission. It is constantly published by the simulation.

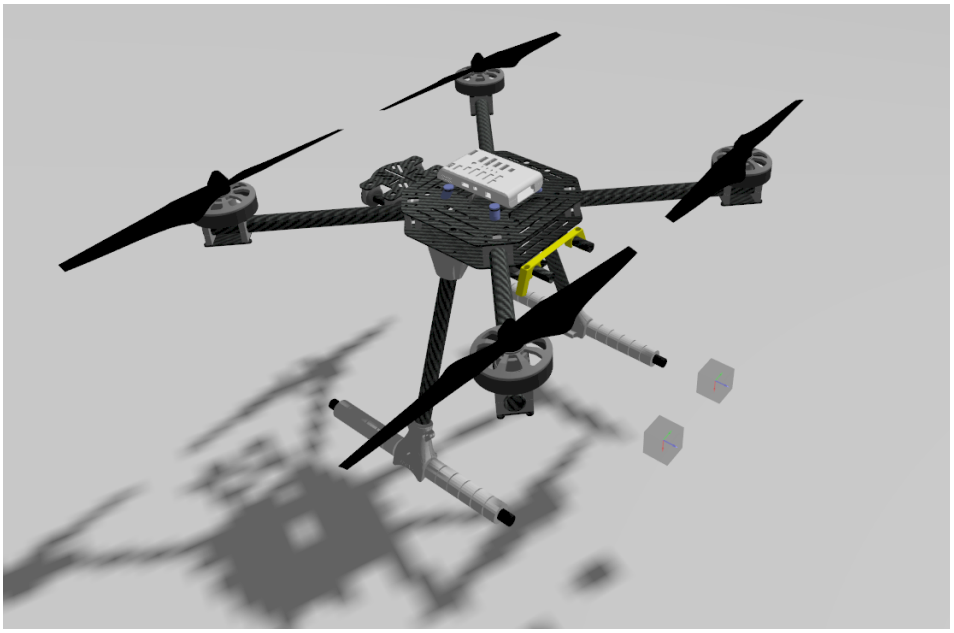


Figure 2.1: Stereo camera on drone indicated by opaque boxes

- The camera pose C

  The camera pose is the frame in which the point cloud is created and aggregated. The relative pose of the cameras are set in the simulated drone's setup file. This transformation is static and is applied to each incoming pose message directly.

Hereafter, the following notation is used:

- ${}_{t_D}C^W$: Transformation from Drone to Camera frame represented in the World frame
- R_DC: Active Rotation from the Drone to the Camera frame
- ${}_{r_D}C^W$: Position vector from the drone to the camera represented in the World frame.

One more thing to note in fig. 2.1 is that in Gazebo's camera convention, the optical axis of a camera points along the x-axis. Therefore, the base link pose was subscribed to and the pose was converted to the camera pose using the adequate transformation. Neither Gazebo's base link nor camera coordinate conventions are relevant as long as we correctly track the pose of a downwards facing camera.

2.1.2 Input Handling

The input of the images as well as the base link pose are received using ROS subscribers. Despite publishing these simulated Gazebo topics at equal rates however, they did not arrive simultaneously. Note that this would be resolved when working with the actual xVIO state estimator as it uses the tracking camera's image and supplies further nodes with that same image as well as the pose with synchronized timestamps.
The pose is only required for the transfer of the created points from the camera frame to the world frame. Therefore, the two input images were processed into a

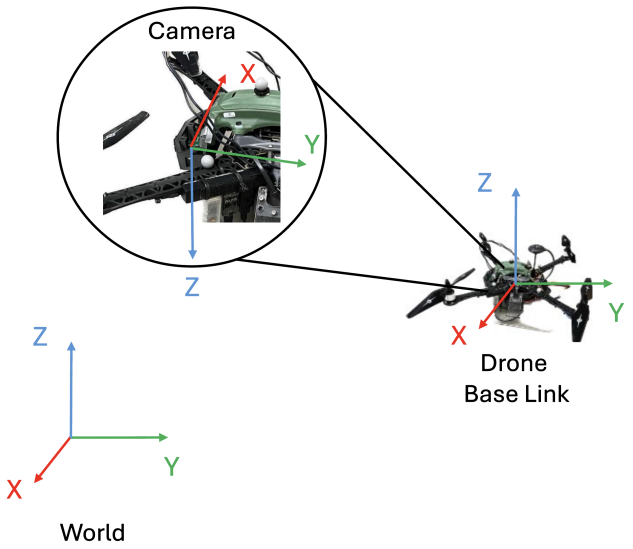


Figure 2.2: Coordinate frames in stereo camera depth setup

depth image in a single step and only after were all three inputs used in order to create the point cloud.

fig. 2.3 schematically shows how the stereo camera depth node handled this shortcoming of asynchronous sensor messages by manually picking the pose which temporally closest corresponds to the image's timestamp. This was possible as the image processing step dominates the computation time of the input handling and the pose can thus be continuously updated in the meantime to best fit the images.

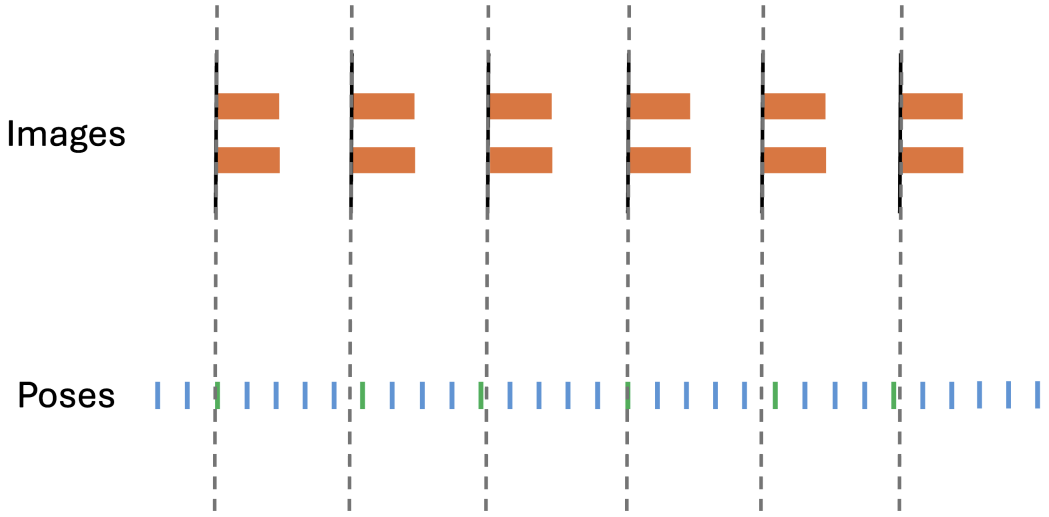


Figure 2.3: Schematic visualization of input synchronization used in the stereo depth node

2.1.3 Disparity Creation

The chosen StereoSGBM (Semi-Global Block Matching) algorithm creates a disparity map from two horizontally aligned images using the following process:

-

StereoSGBM vs StereoBM

Apart from the StereoSGBM implementation, OpenCV provides the StereoBM algorithm which in general is faster but less precise.
Implementing and comparing the two algorithms the following results were seen:
As can be seen in the above figure, StereoBM sometimes showed unsatisfactory point clouds. In general, a swift algorithm like StereoBM is preferable for our purposes. Nevertheless, precise terrain reconstruction is a vital component of the pipeline, and thus, the quality must not be compromised. Therefore, StereoSGBM was retained as the algorithm of choice for the stereo depth creation.

2.1.4 Point Cloud Creation

Having created a disparity image using the approach laid out in section 2.1.3, the disparity pixels are first converted to the depth values using the classic disparity depth relation 1.1 shown in section 1.1.2.
From the created depth image, the pose and the camera parameters, the 3D locations of each detected point can be derived. This is done by simply tracing the projection line of the detected point as indicated in fig. 2.4 and using the similar triangles as shown in fig. 2.5

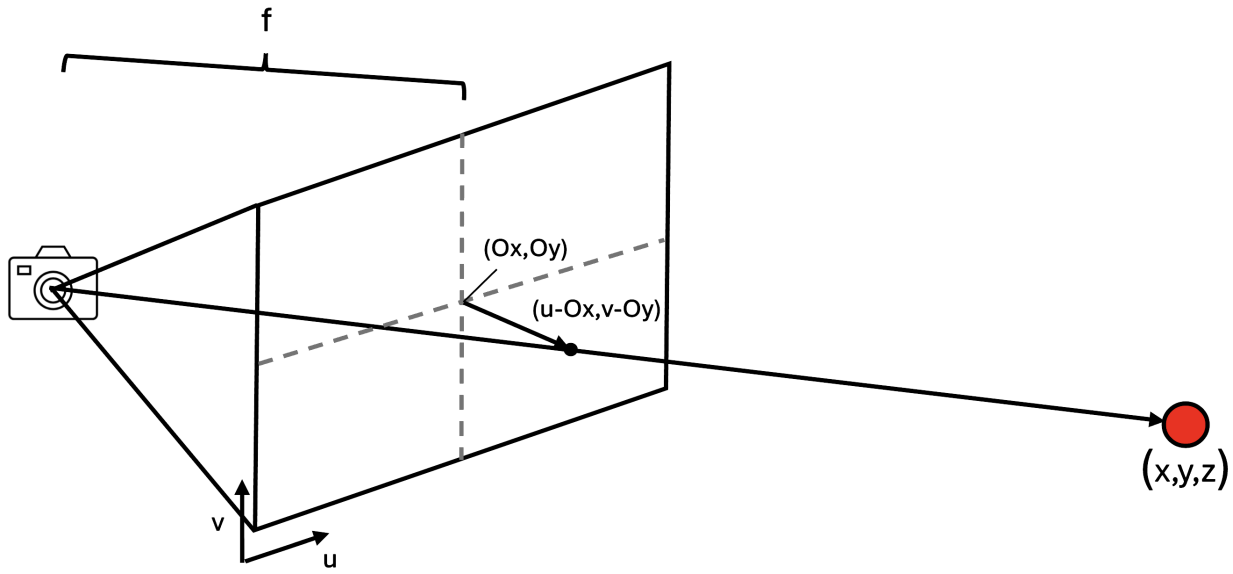


Figure 2.4: Schematic of the line projection procedure to derive the 3D location of detected points

$y = z \quad = z$ the depth value is already stored in the depth image and can be retained directly. (2.1)

(2.2)

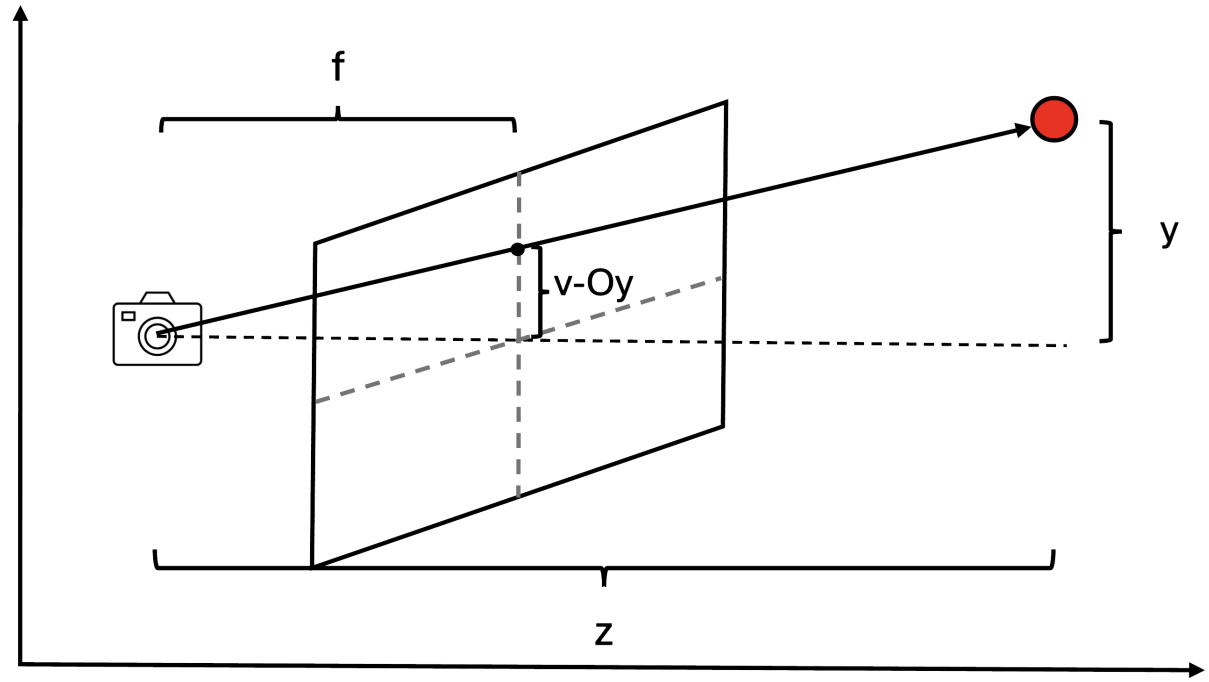


Figure 2.5: Schematic of the line projection procedure to derive the 3D location of detected points

The final output of the node is a generated point cloud in the world frame together with two poses representing the camera locations of the generated point cloud.

2.1.5 Switching

In order to achieve the final desired perception mechanism of flying laterally with SFM and using a stereo camera depth node at low altitudes, one needs to switch between the two alternatives.
The obvious decision to use in the switching mechanism is the drone's current altitude above ground. This could be achieved by analyzing the generated point cloud at a given iteration to determine the median altitude which indicates the altitude above ground. This however is avoidable computational overhead.
As mentioned in **??** the drone has a laser range finder on board. This allows us to get an estimate of the altitude above ground at any given moment without the need for image processing.
Therefore, the switching is performed by using a separate ROS subscriber which continuously checks the LRF's measurement and activates or deactivates the SFM node and stereo node respectively.

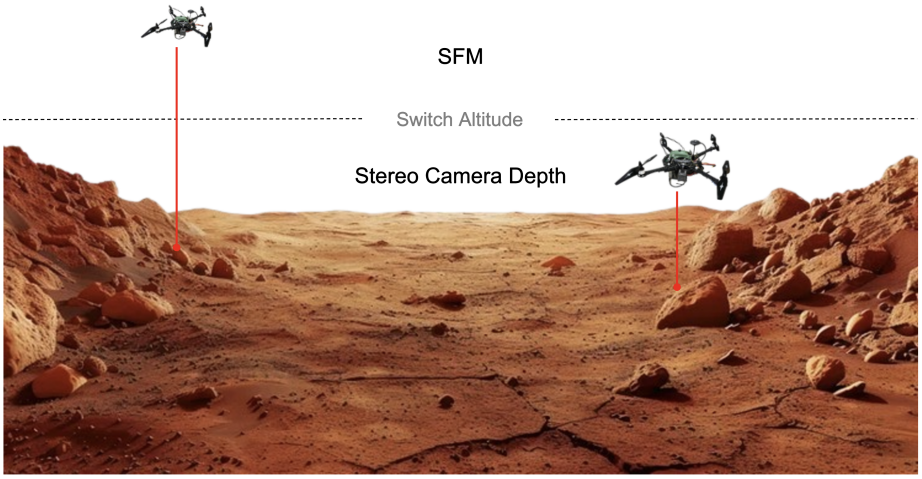


Figure 2.6: Laser Ranger Finder Based Switch between Depth Sources

2.1.6 Landing Site Detection without Lateral Motion

Taking off vertically with the drone in the simulation, the first landing site without lateral motion was found.

Figure 2.7: Drone during vertical ascent in simulation

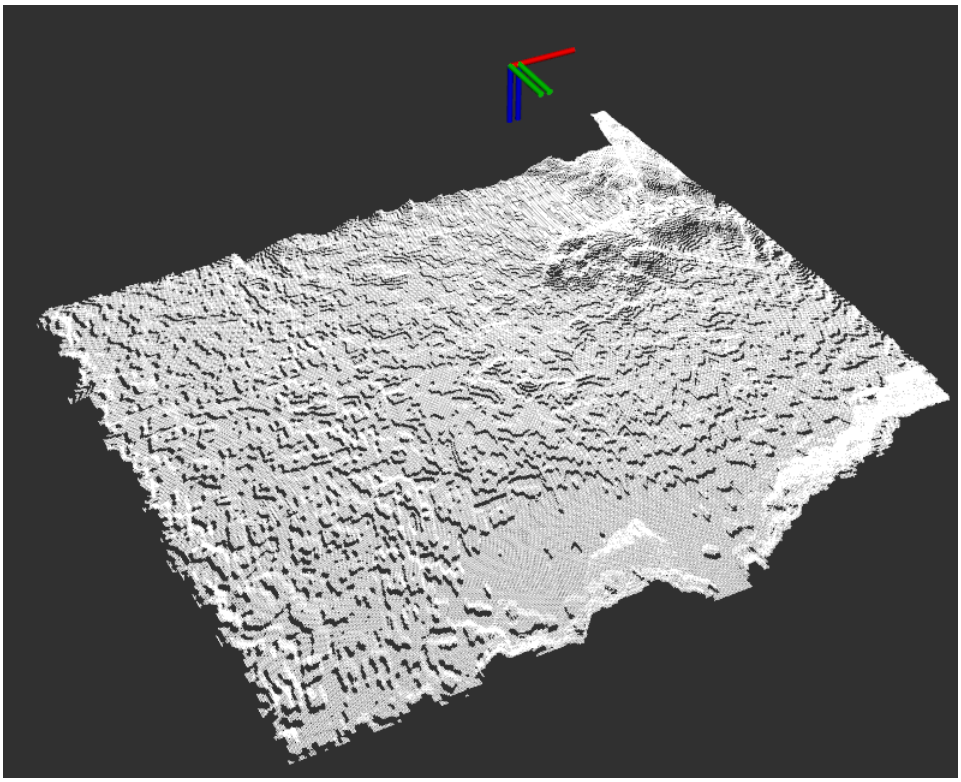


Figure 2.8: RViz visualization of created point cloud from stereo camera

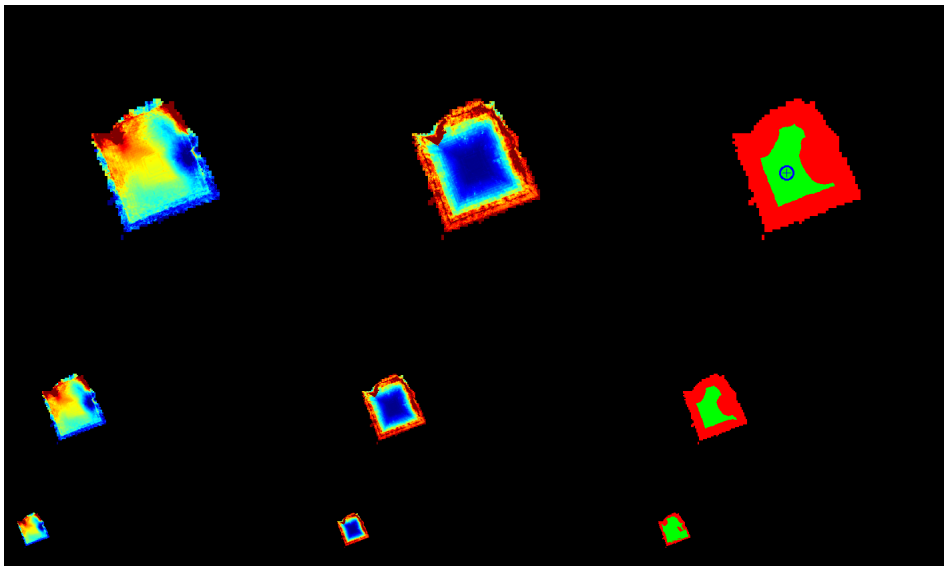


Figure 2.9: LSD Debug output displaying LORNA's first detected landing site during vertical motion

2.2 Qualitative Practical Analysis

Once implemented the landing site detection instance could be supplied by the stereo depth node. The result thereof can be seen below:

Figure 2.10: Considered terrain patch in Gazebo simulation

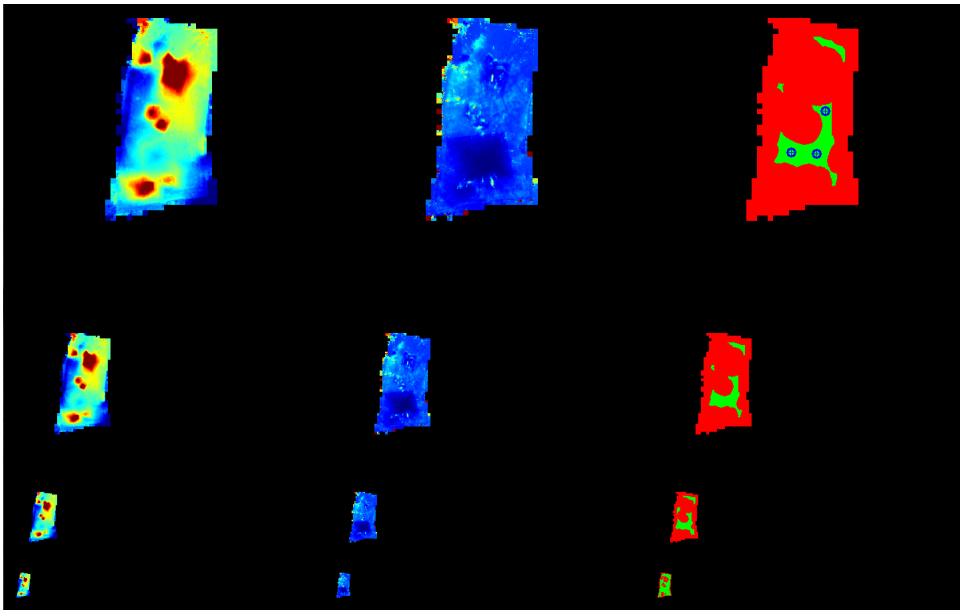


Figure 2.11: Stereo camera depth supplied LSD debug image at 2.5 m altitude

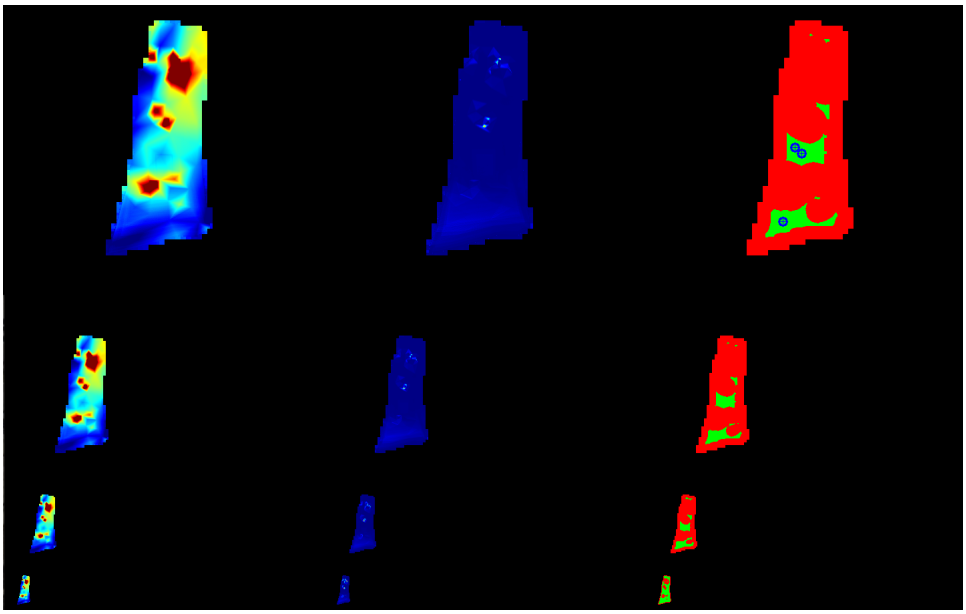


Figure 2.12: GT depth supplied LSD debug image at 2.5 m altitude

When comparing the result to the ground truth LSD output it can be seen that LSD creates a very accurate DEM from the stereo camera depth input. The landing sites detected are reasonable when compared to the terrain reference.
As the stereo camera has a relatively small, fixed baseline. The usage domain is restricted to low altitudes. The residual part of a mission is still flown using SFM.

Bibliography

[1] L. Di Pierno, R. Brockers, and R. Hewitt, "Autonomous Long-Range Flight Execution for Future Mars Rotorcraft," 2024.

[2] H. Hirschmuller, "Stereo processing by semiglobal matching and mutual information," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 30, no. 328-341, 2008.

[3] M. Domnik, P. Proenca, J. Delaune, J. Thiem, and R. Brockers, "Dense 3D-Reconstruction from Monocular Image Sequences for Computationally Constrained UAS," 2021.