Monocular Visual Odometry from Frame to Frame Intensity Differences for Planetary Exploration Mobile Robots

Geovanni Martinez

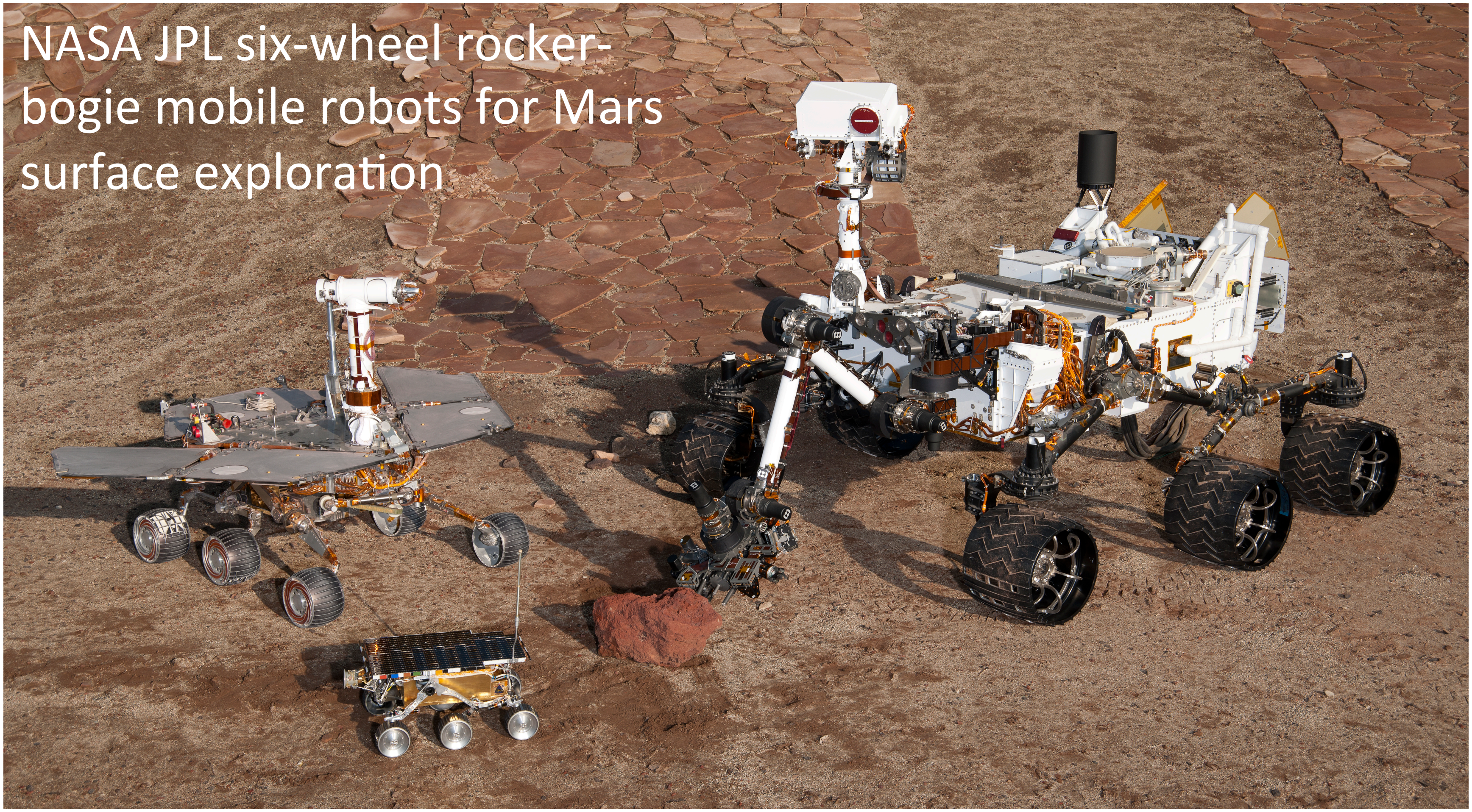
Image Processing and Computer Vision Research Laboratory (IPCV-LAB)
School of Electrical Engineering, University of Costa Rica

IEEE Workshop on Robot Vision, 16-17 January 2013, Tampa, Florida, USA

Overview

- Introduction
- Problem
- Current approach
- Our approach
- Results
- Summary

Introduction



Courtesy NASA/JPL-Caltech

Introduction

- These rovers must be able to autonomously navigate to the science targets
- Any navigation error can cost the loss of the entire day of scientific activity
- For precise autonomous navigation, the rovers must know precisely its position and orientation at any time

Introduction

- The rover´s position and orientation are obtained by integrating its motion **B** over time, i.e. by integrating its translation Δ**T** and rotation Δ**Ω**, over time, respectively
- Δ**Ω** is estimated from measurements of three gyros provided by an IMU onboard the rover
- Δ**T** is estimated from encoder readings of how much the wheels turned (wheel odometry)

Problem

- Excessive wheel slippage on steep slopes and soft soils
- This causes large errors particularly on the estimated rover's position from wheel odometry

Courtesy NASA/JPL-Caltech

Current Approach

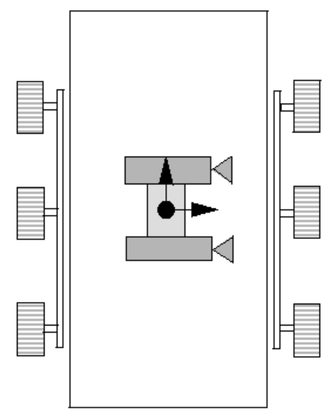
- Estimate also the rover´s motion **B** by applying a feature based stereo visual odometry algorithm
- Correct any position error by using the motion estimates $\hat{\mathbf{B}}$ provided by the stereo visual odometry algorithm

Stereo Visual Odometry

The stereo visual odometry algorithm estimates the rover's motion **B** from 3D correspondences (3D offsets) between two sets of 3D feature point positions, which were previously obtained from two consecutive stereo image pairs captured before and after the rover´s motion, respectively, by a stereo video camera attached rigidly to the rover

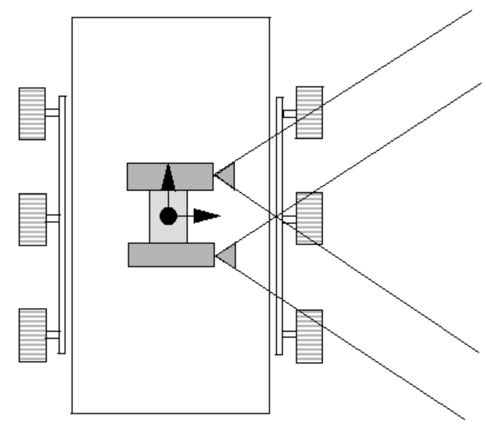
Stereo Visual Odometry

Algorithm



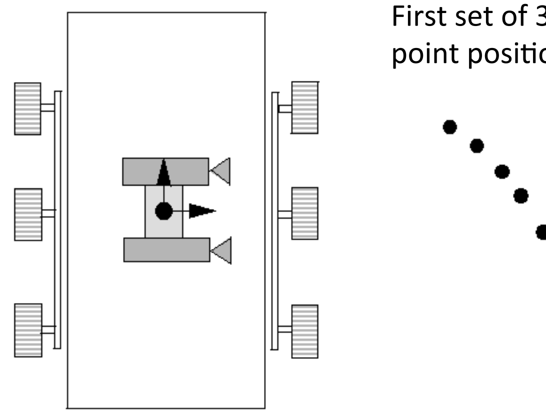
Stereo Visual Odometry

1. Capture a first stereo image pair before the rover's motion



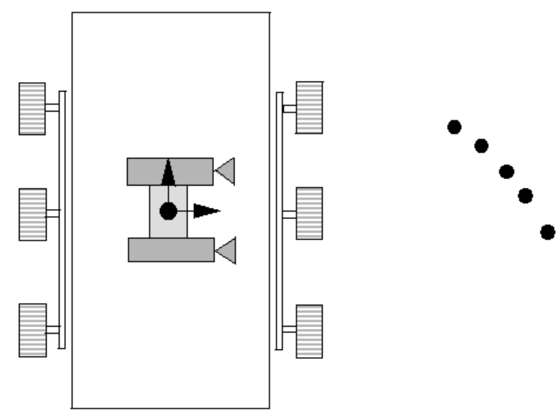
Stereo Visual Odometry

2. Estimate the 3D positions of a first set of feature points from the first stereo image pair by using stereo triangulation



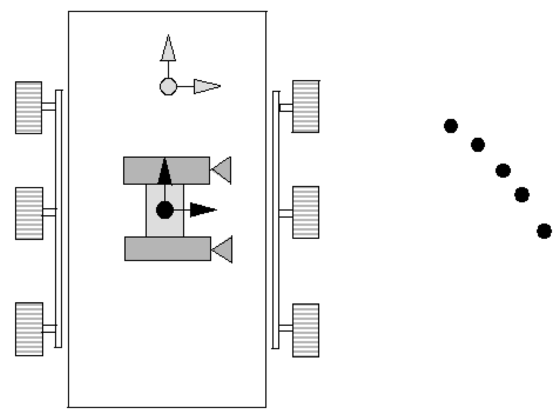
Stereo Visual Odometry

3. Allow the rover to move



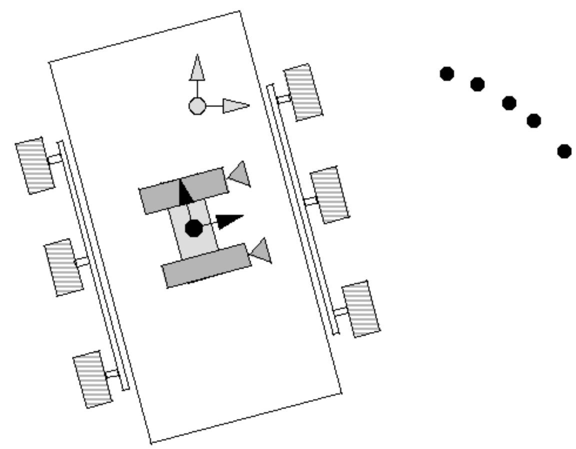
Stereo Visual Odometry

3. Allow the rover to move



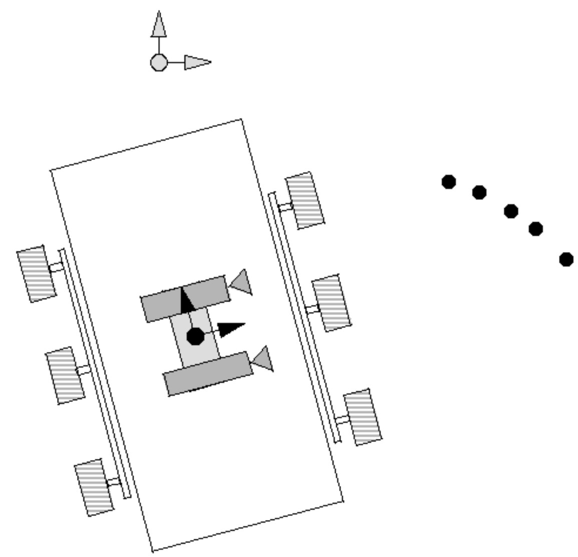
Stereo Visual Odometry

3. Allow the rover to move



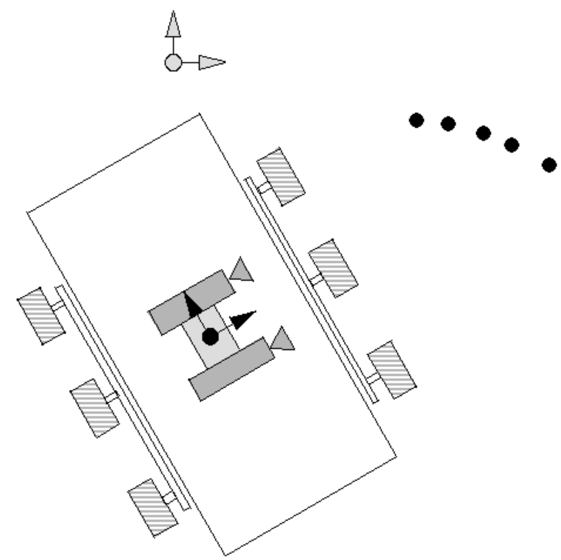
Stereo Visual Odometry

3. Allow the rover to move



Stereo Visual Odometry

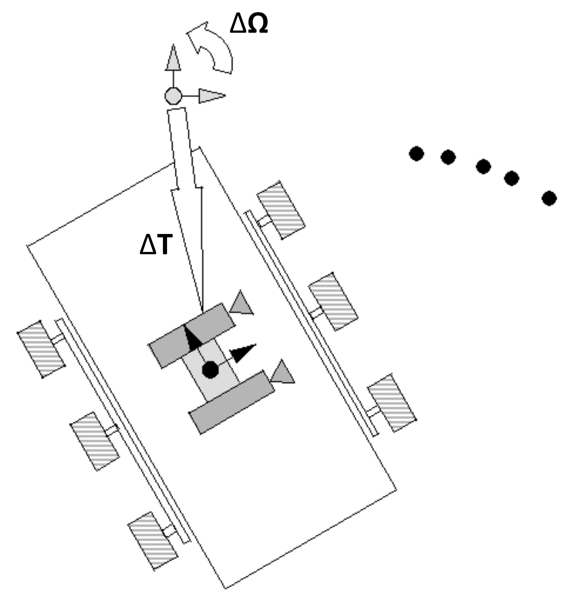
3. Allow the rover to move



Stereo Visual Odometry

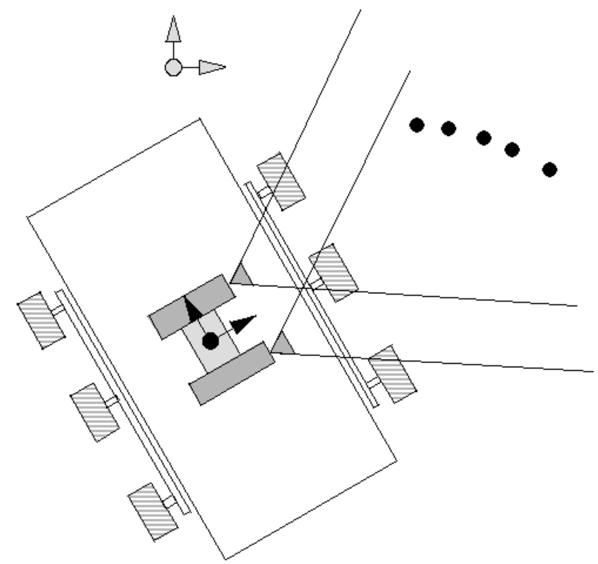
3. Allow the rover to move

   The rover´s motion is described by a rotation $\Delta\boldsymbol{\Omega}=(\Delta\omega_X, \Delta\omega_Y, \Delta\omega_Z)^T$ and then a translation $\Delta\mathbf{T}=(\Delta T_X, \Delta T_Y, \Delta T_Z)^T$, where $\mathbf{B}=(\Delta T_X, \Delta T_Y, \Delta T_Z, \Delta\omega_X, \Delta\omega_Y, \Delta\omega_Z)^T$



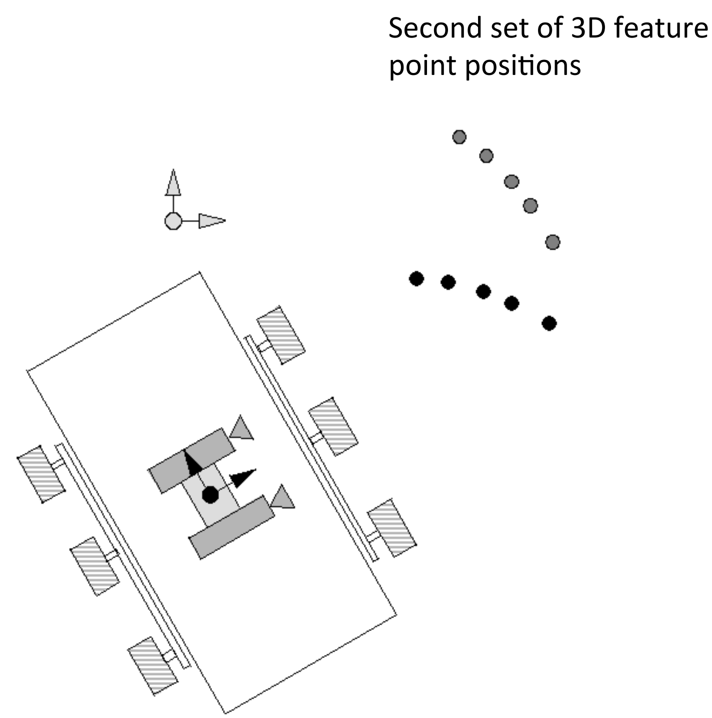
Stereo Visual Odometry

4. Capture a second stereo image pair after rover´s motion



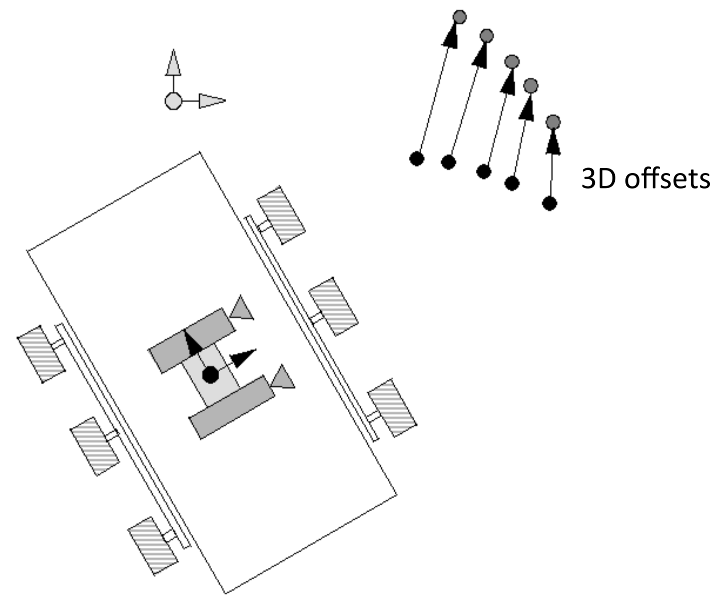
Stereo Visual Odometry

5. Estimate the 3D positions of a second set of feature points from the second stereo image pair by using stereo triangulation



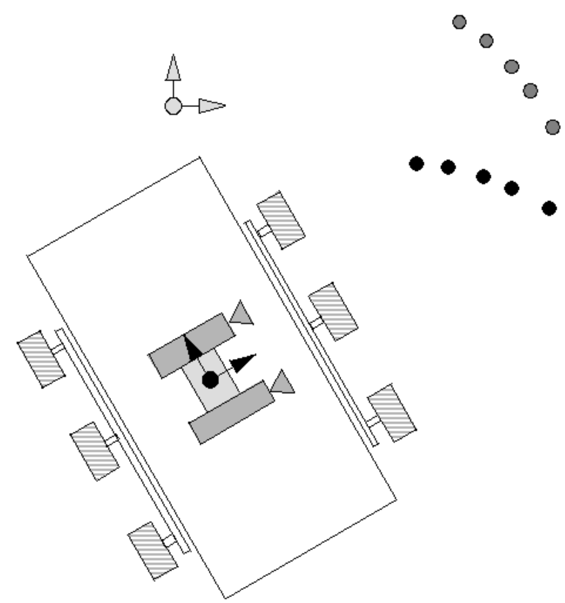
Stereo Visual Odometry

6. Establish the 3D correspondences (3D offsets) between the two sets of 3D feature point positions before and after the rover's motion



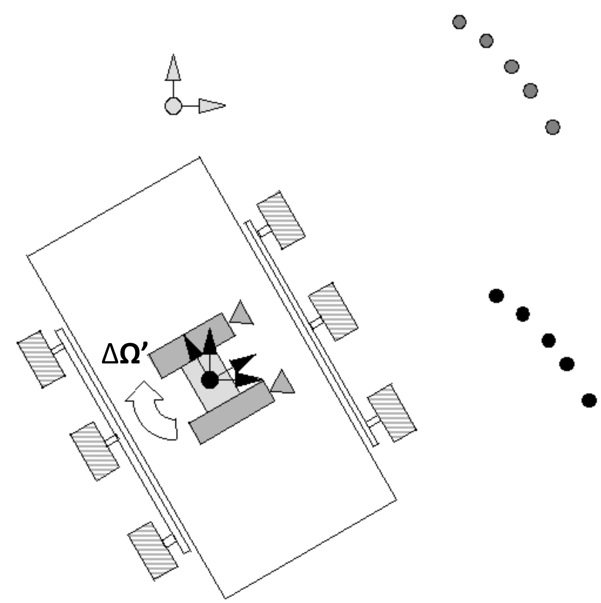
Stereo Visual Odometry

7. Search for those motion parameters **B'** that move rigidly the first set of 3D features point positions (before rover´s motion) to that place where the 3D offsets become as small as possible



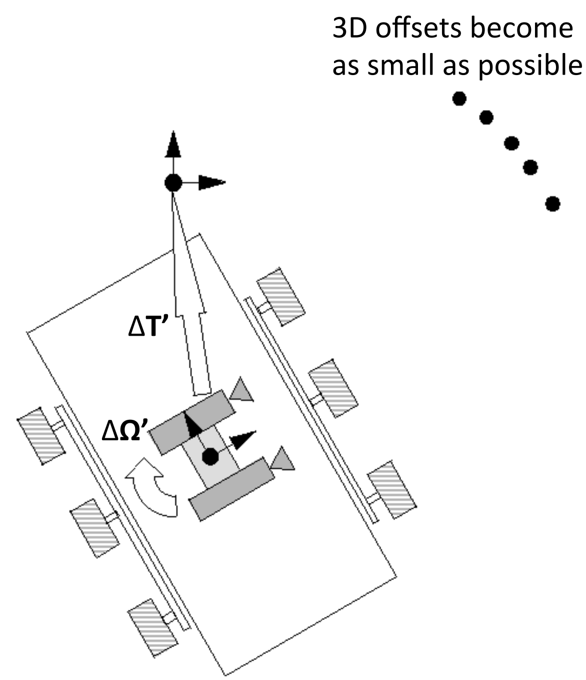
Stereo Visual Odometry

7. Search for those motion parameters **B'** that move rigidly the first set of 3D features point positions (before rover´s motion) to that place where the 3D offsets become as small as possible



Stereo Visual Odometry

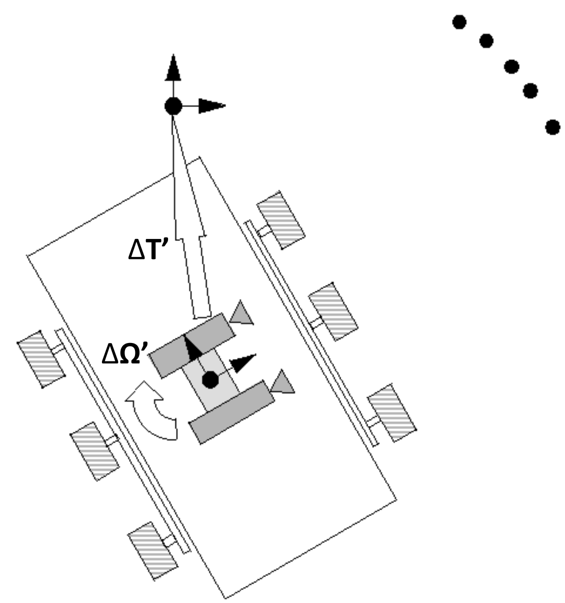
7. Search for those motion parameters **B’** that move rigidly the first set of 3D features point positions (before rover´s motion) to that place where the 3D offsets become as small as possible



The parameters **B’** are searched by maximizing a likelihood function of the established 3D correspondences

Stereo Visual Odometry

7. Search for those motion parameters **B’** that move rigidly the first set of 3D features point positions (before rover´s motion) to that place where the 3D offsets become as small as possible



The rover’s motion estimates are $\hat{\mathbf{B}}$=-**B’**

Our Approach

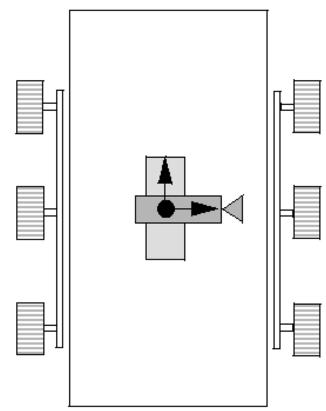
- Estimate also the rover´s motion **B** by applying a monocular visual odometry algorithm based on intensity differences at observation points
- Correct any position error by using the motion estimates $\hat{\mathbf{B}}$ provided by the monocular visual odometry algorithm

Monocular Visual Odometry

The monocular visual odometry algorithm estimates the rover's motion **B** from the intensity differences at observation points between two consecutive intensity images, which were captured before and after the rover´s motion, respectively, by a monocular video camera attached rigidly to the rover

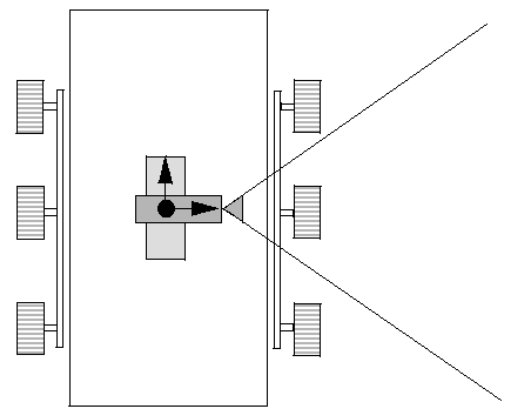
Monocular Visual Odometry

Algorithm



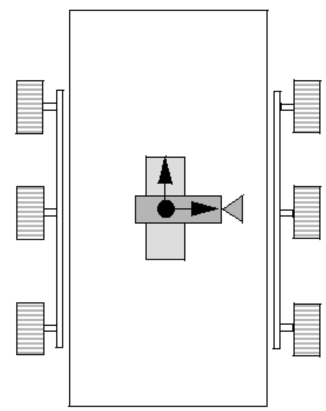
Monocular Visual Odometry

1. Capture a first intensity image before the rover's motion

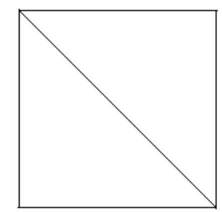


Monocular Visual Odometry

2. Adapt the size, position and orientation of a generic surface model to the content of the first intensity image



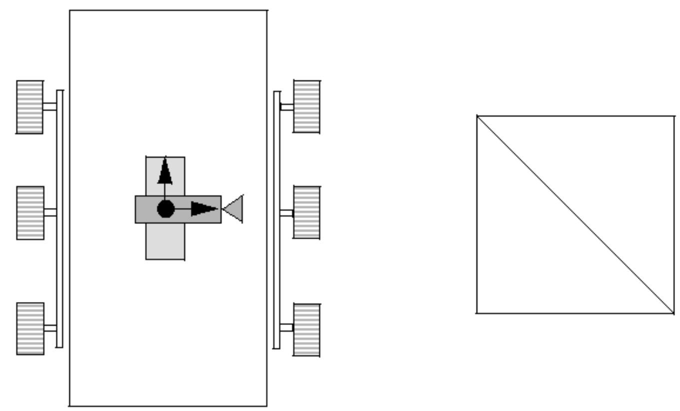
Adapted surface model



In this contribution the generic surface model is a rigid and flat mesh of triangles consisting of only two triangles

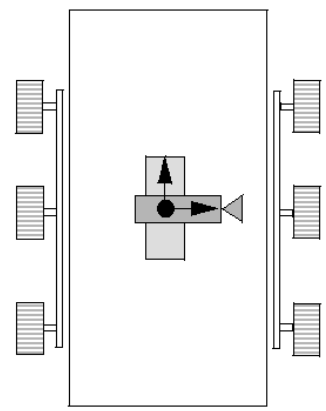
Monocular Visual Odometry

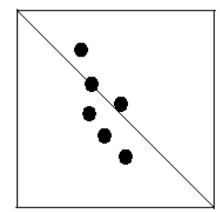
3. Select as observation points those image points in the first intensity image with high linear intensity gradients



Monocular Visual Odometry

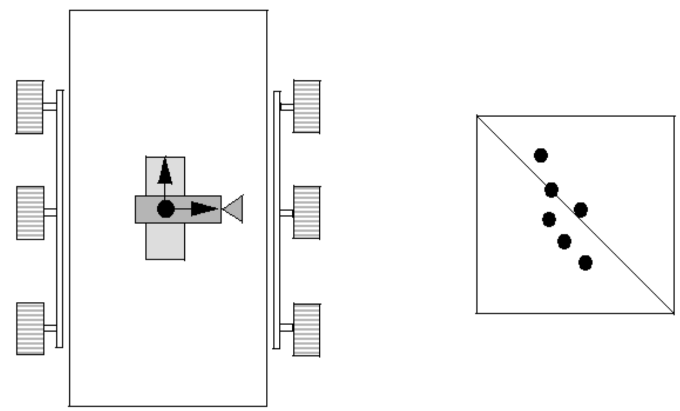
3. Select as observation points those image points in the first intensity image with high linear intensity gradients and attach them (together with their intensity values) rigidly to the surface model





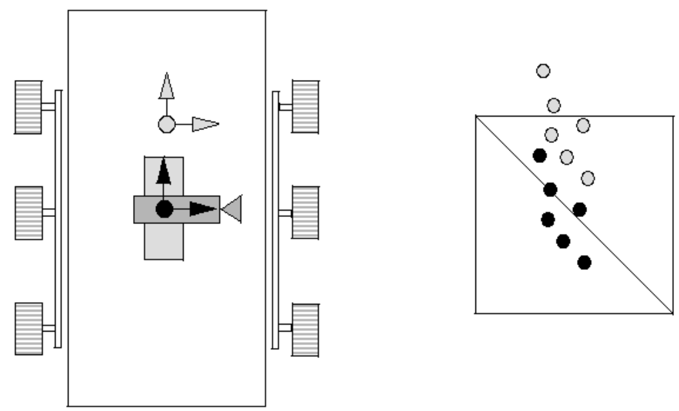
Monocular Visual Odometry

4. Allow the rover to move



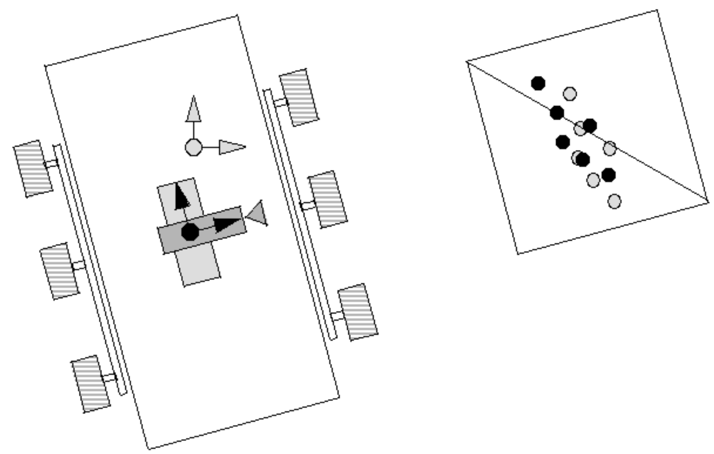
Monocular Visual Odometry

4. Allow the rover to move



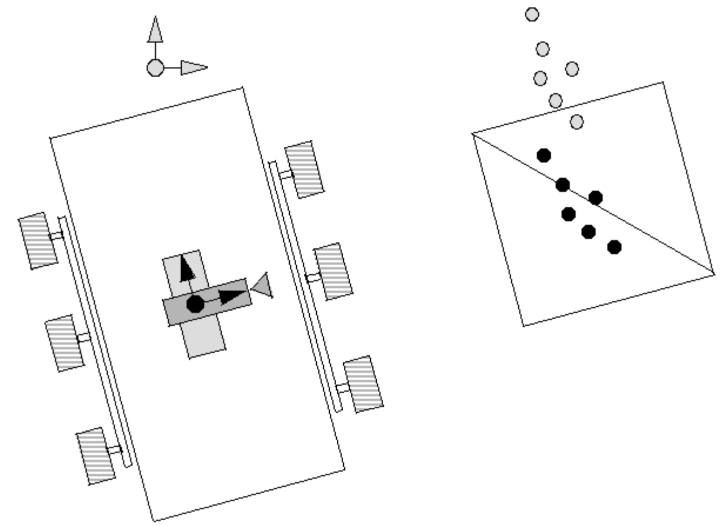
Monocular Visual Odometry

4. Allow the rover to move



Monocular Visual Odometry

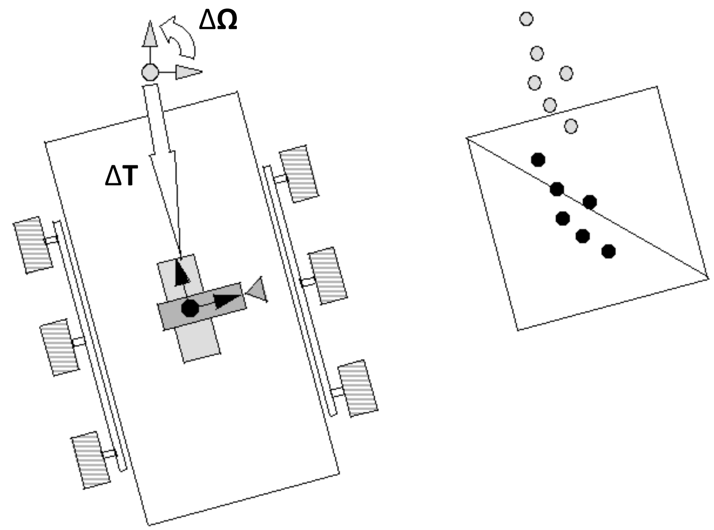
4. Allow the rover to move



Monocular Visual Odometry

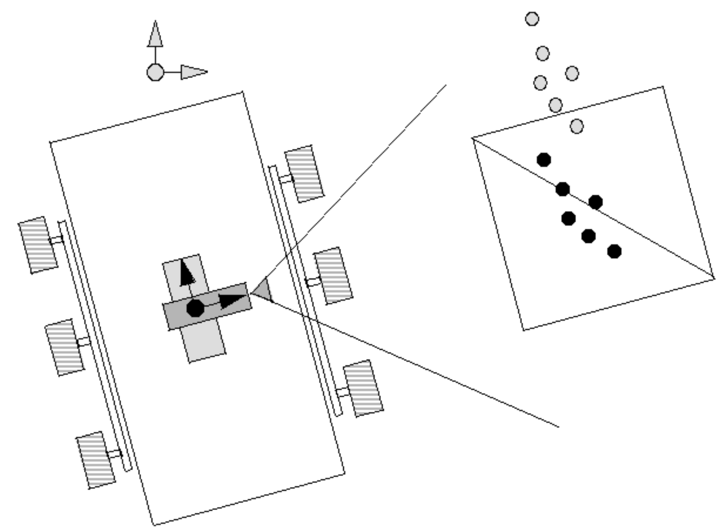
4. Allow the rover to move

   The rover´s motion is described by a rotation $\Delta\mathbf{\Omega}=(\Delta\omega_X, \Delta\omega_Y, \Delta\omega_Z)^T$ and then a translation $\Delta\mathbf{T}=(\Delta T_X, \Delta T_Y, \Delta T_Z)^T$, where $\mathbf{B}=(\Delta T_X, \Delta T_Y, \Delta T_Z, \Delta\omega_X, \Delta\omega_Y, \Delta\omega_Z)^T$



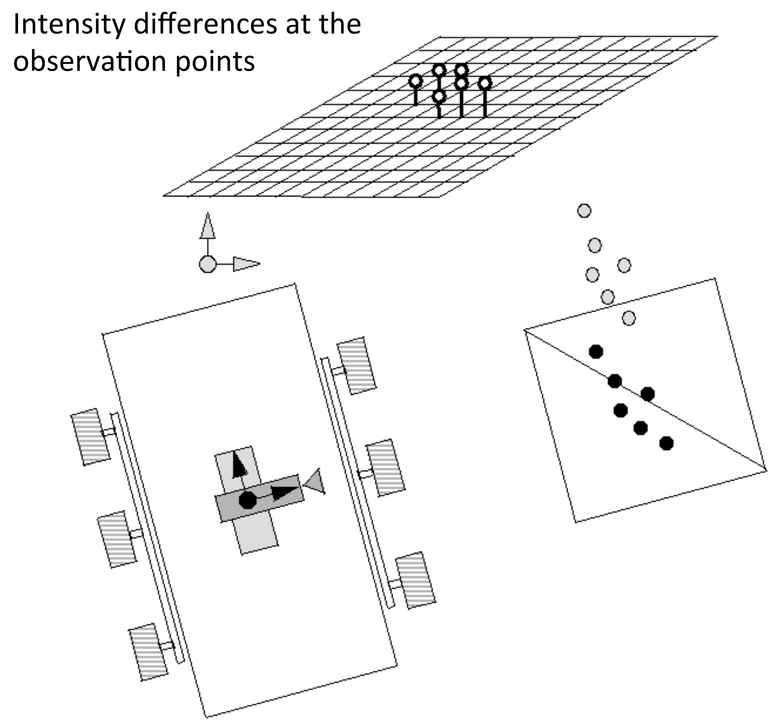
Monocular Visual Odometry

5. Capture a second intensity image after rover´s motion



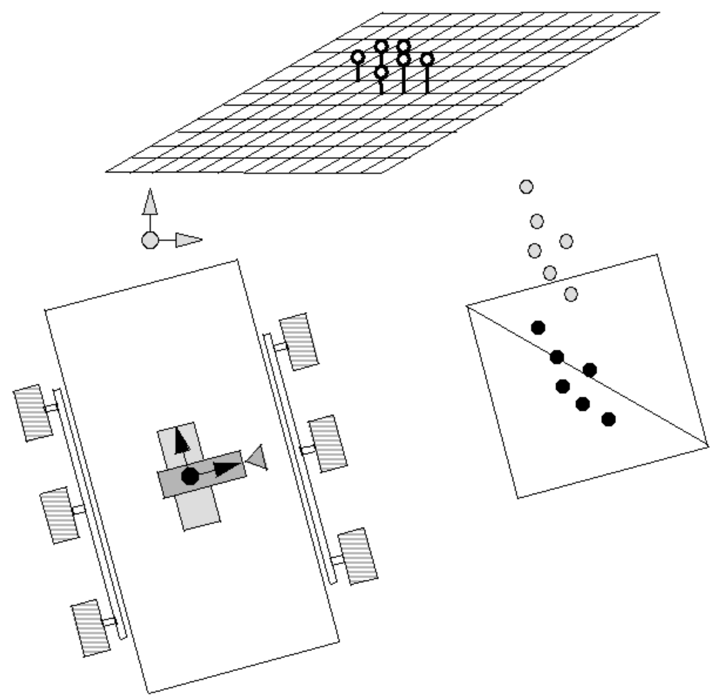
Monocular Visual Odometry

6. Project the observation points into the image plane and compute the intensity differences between their intensity values and the second intensity image



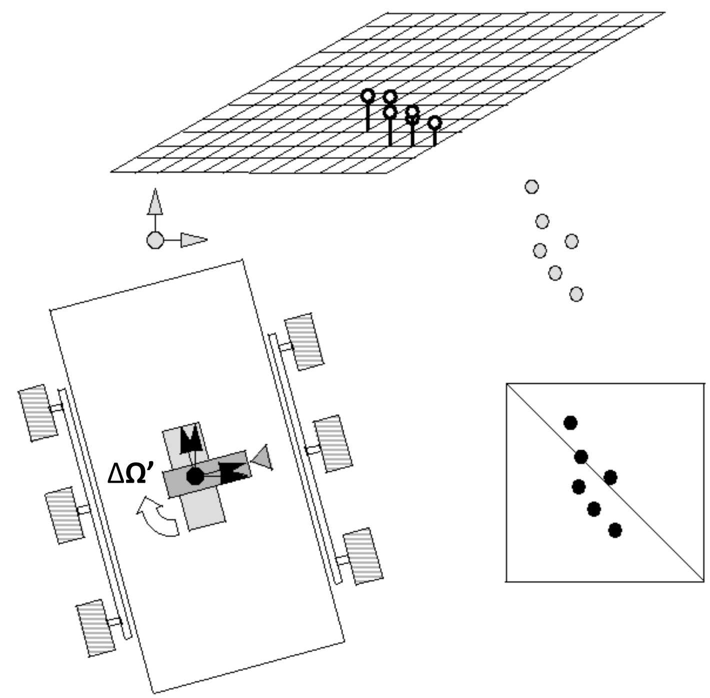
Monocular Visual Odometry

7. Search for those parameters **B'** that move the surface model (and therefore the rigidly attached observation points) to that place where the intensity differences become as small as possible



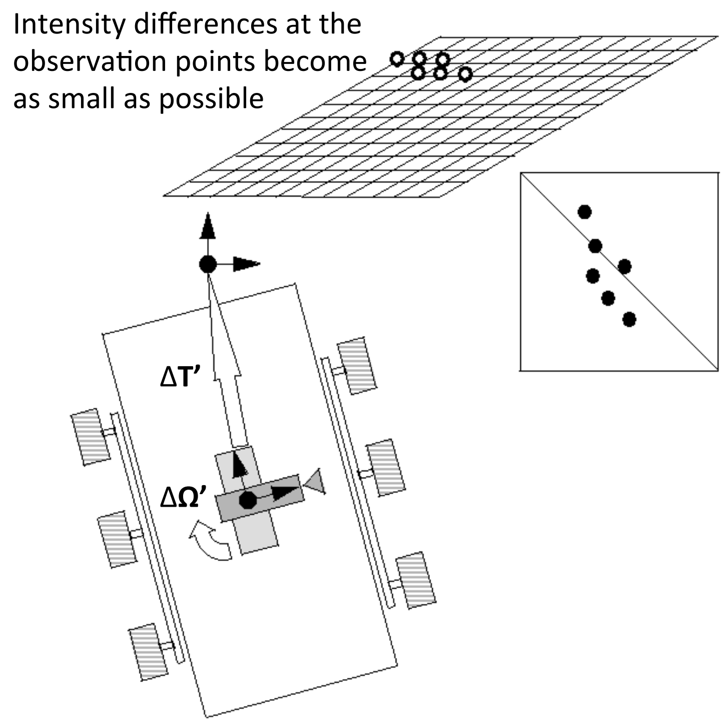
Monocular Visual Odometry

7. Search for those parameters **B'** that move the surface model (and therefore the rigidly attached observation points) to that place where the intensity differences become as small as possible



Monocular Visual Odometry

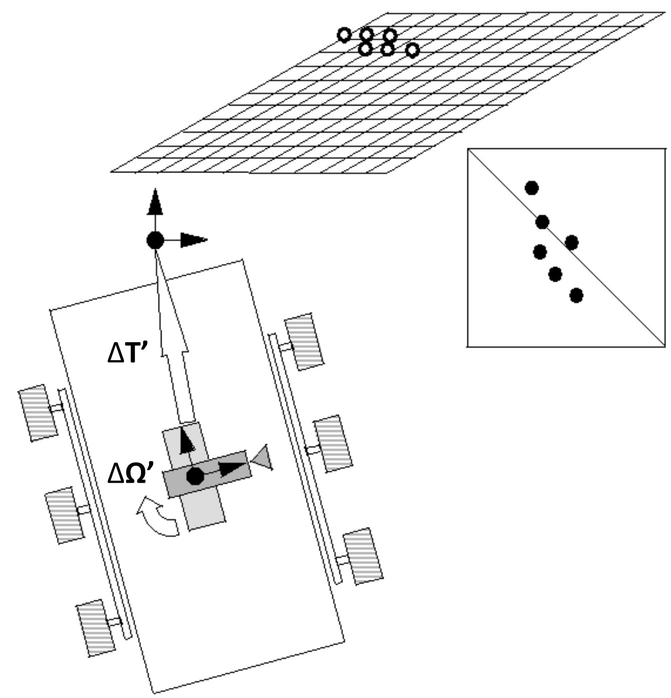
7. Search for those parameters **B'** that move the surface model (and therefore the rigidly attached observation points) to that place where the intensity differences become as small as possible



The parameters **B'** are searched by maximizing a likelihood function of the intensity differences at the observation points

Monocular Visual Odometry

7. Search for those parameters **B’** that move the surface model (and therefore the rigidly attached observation points) to that place where the intensity differences become as small as possible



The rover’s motion estimates are $\hat{\mathbf{B}}$=-**B’**

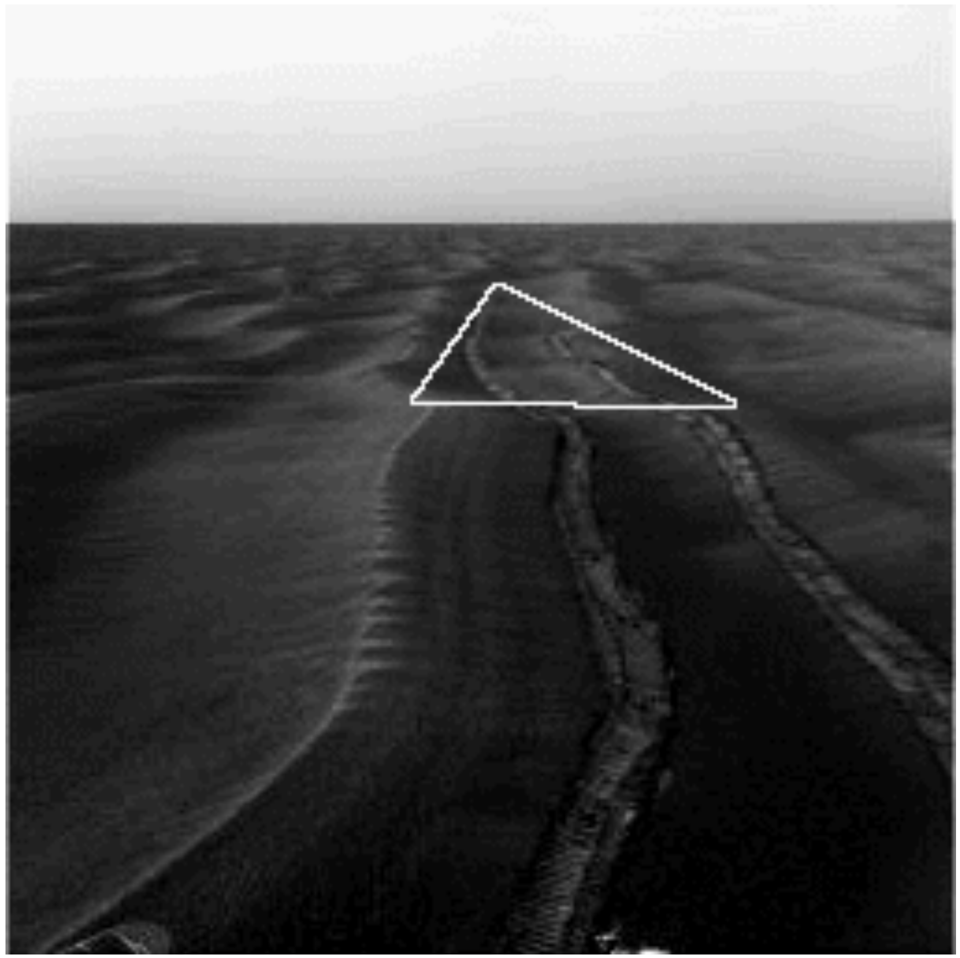
Experimental Results

- Implemented in C under Mac OS X
- Experiments performed on an iMac with an Intel Core i5 at 3.1 GHz and 12 GB RAM
- Tested with both noisy synthetic image sequences and real image sequences
- Real Image sequences captured by the left navigation camera of the Mars Exploration Rover (MER) Opportunity at different Martian landscapes with image size of 256x256 $pixel^2$

Experimental Results

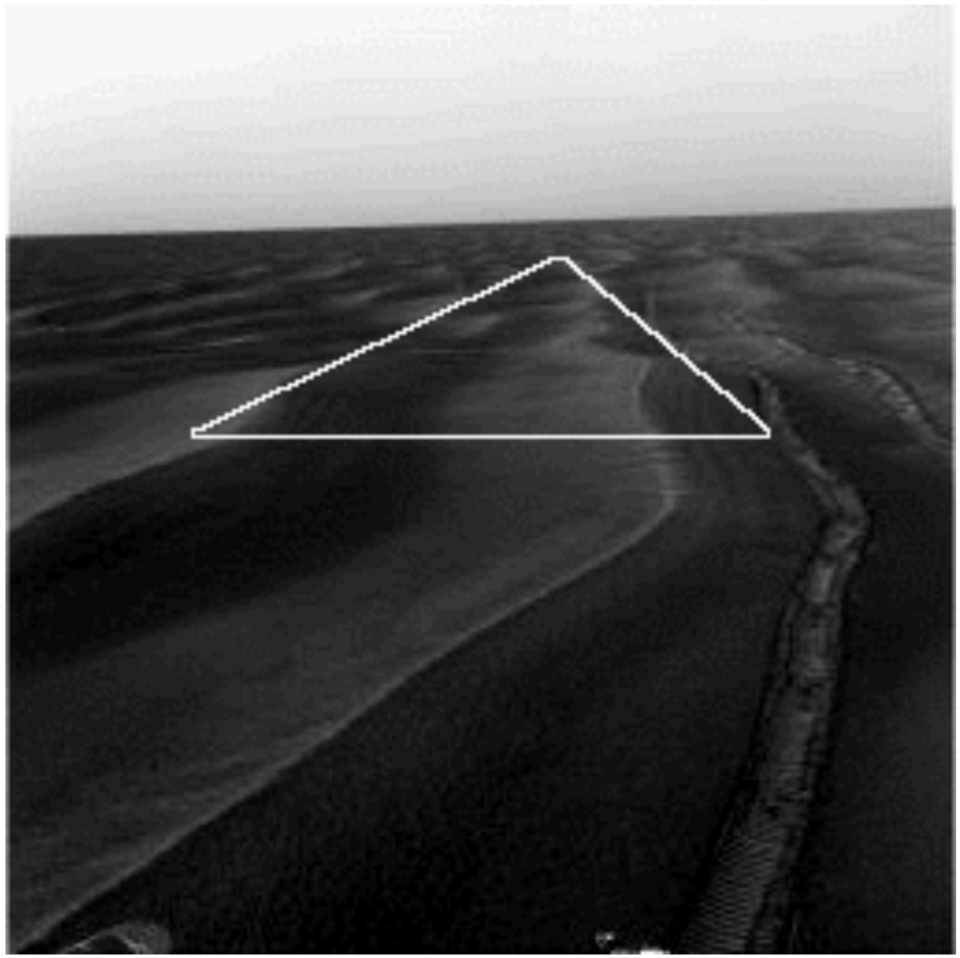
- High accuracy and reliability with the noisy synthetic image sequences
- Average processing time of 0.1 sec/image for the real image sequences
- Tracking was never lost

Experimental Results



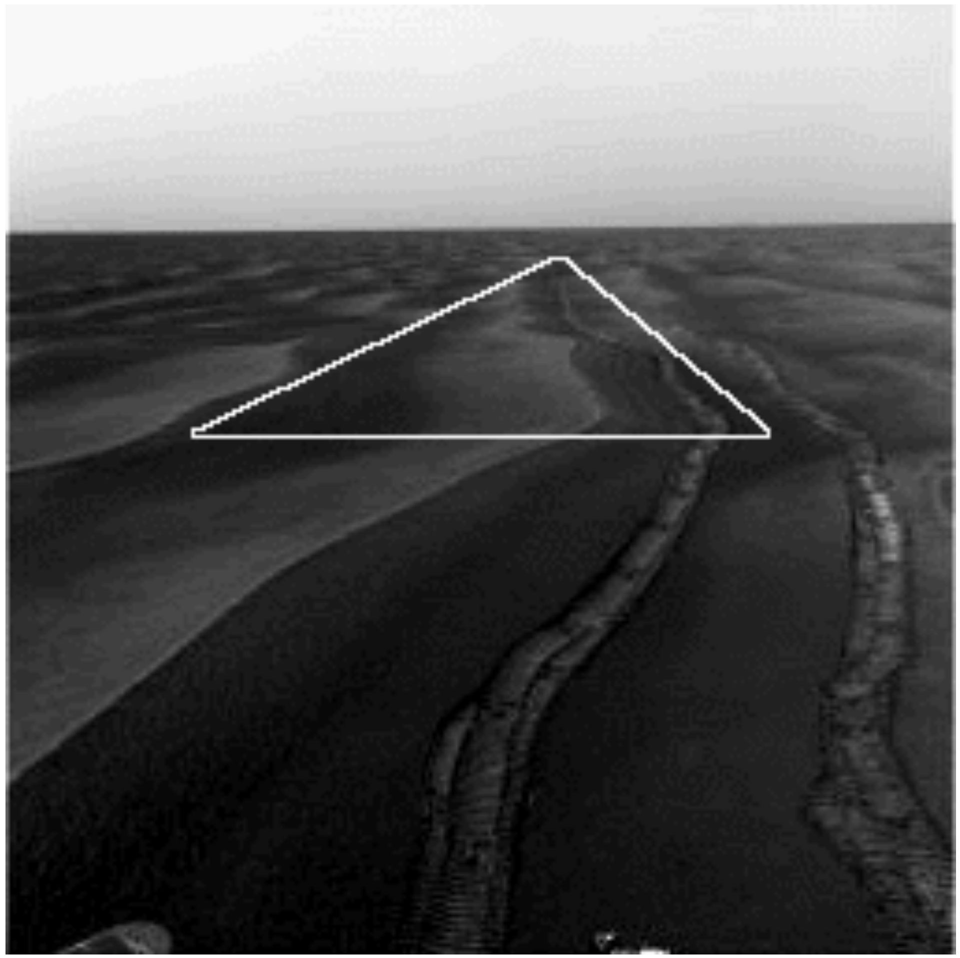
Original image sequence courtesy of NASA/JPL-Caltech

Experimental Results



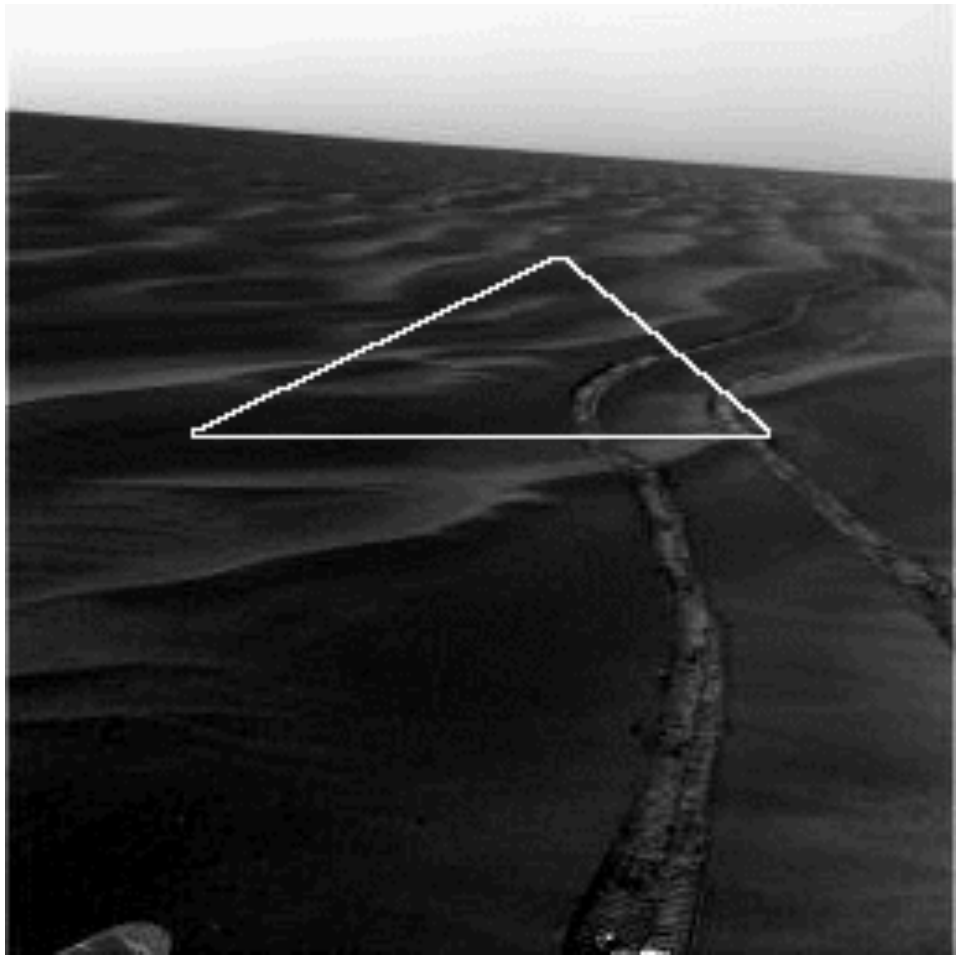
Original image sequence courtesy of NASA/JPL-Caltech

Experimental Results



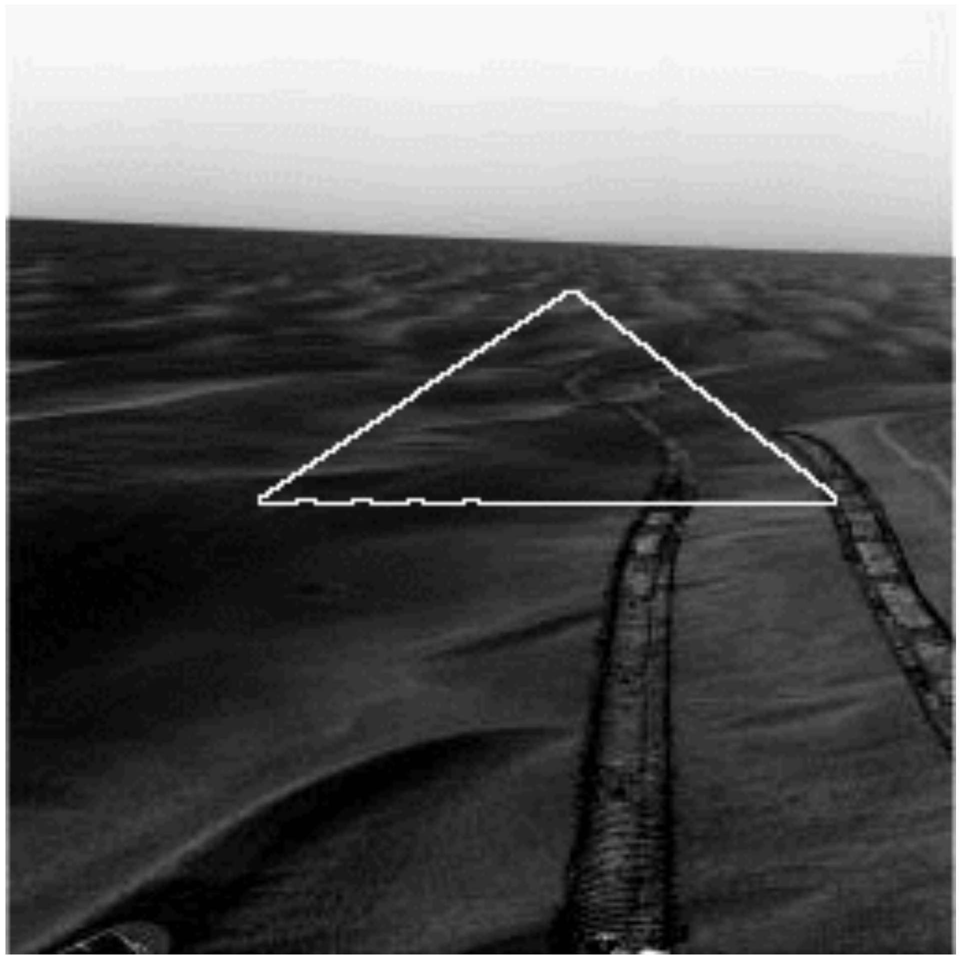
Original image sequence courtesy of NASA/JPL-Caltech

Experimental Results



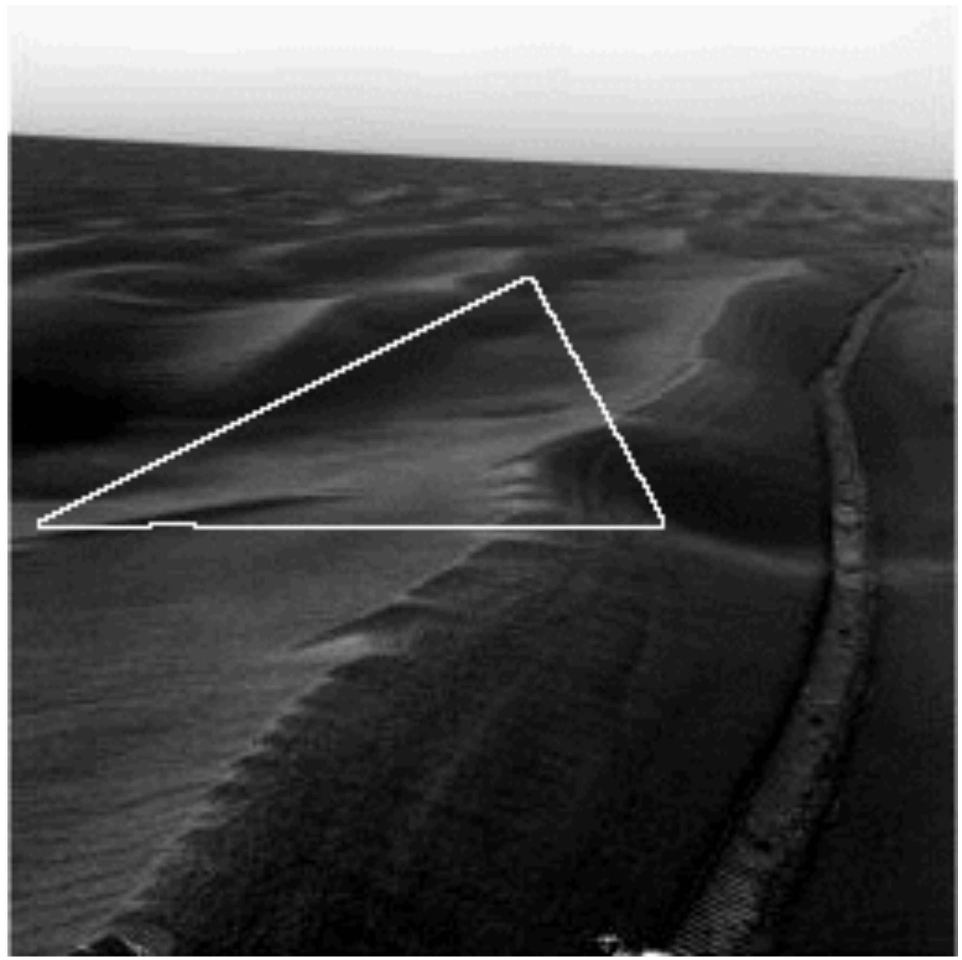
Original image sequence courtesy of NASA/JPL-Caltech

Experimental Results



Original image sequence courtesy of NASA/JPL-Caltech

Experimental Results



Original image sequence courtesy of NASA/JPL-Caltech

Summary

- Implemented and tested a prototype for monocular visual odometry based on intensity differences
- It requires the adaptation of the size and pose of a generic surface model only once at the beginning of the image sequence
- Average processing time of 0.1 sec/image
- Since it operates just with a single monocular video camera, it might weight less, as well as require less energy and space than stereo visual odometry
- It could be merged with stereo visual odometry by using sensor fusion to improve long range autonomous navigation
- Validation of its performance on a real rover test bed is still missing!

Thank you!