Real-Time Robot 3D Pose Computation from NIR Imagery and ToF Depth Maps for Space Applications

Geovanni Martínez

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

October 5th, 2023

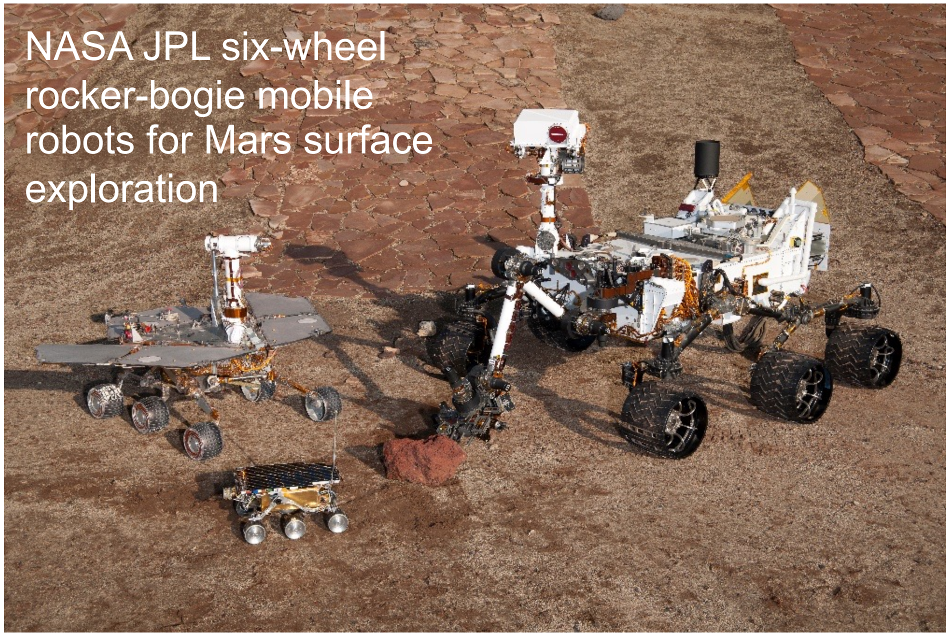
iCASAT-2023, October 04-06, 2023, Queretaro, Mexico

Overview

- Introduction
- Approach
- Algorithm
- Results
- Summary

Introduction

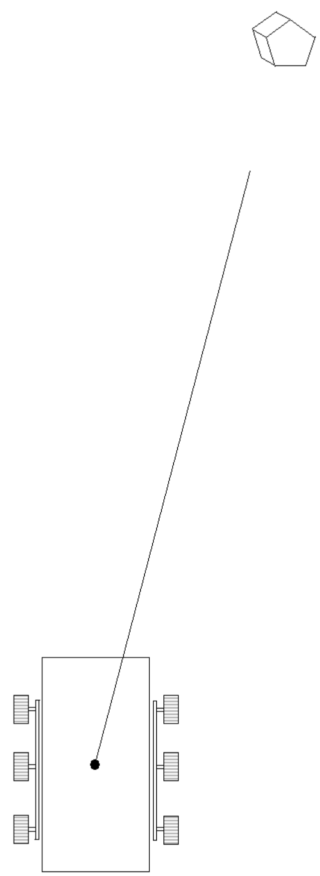
- One important feature of these planetary exploration robots is their ability to navigate autonomosly



Courtesy NASA/JPL-Caltech

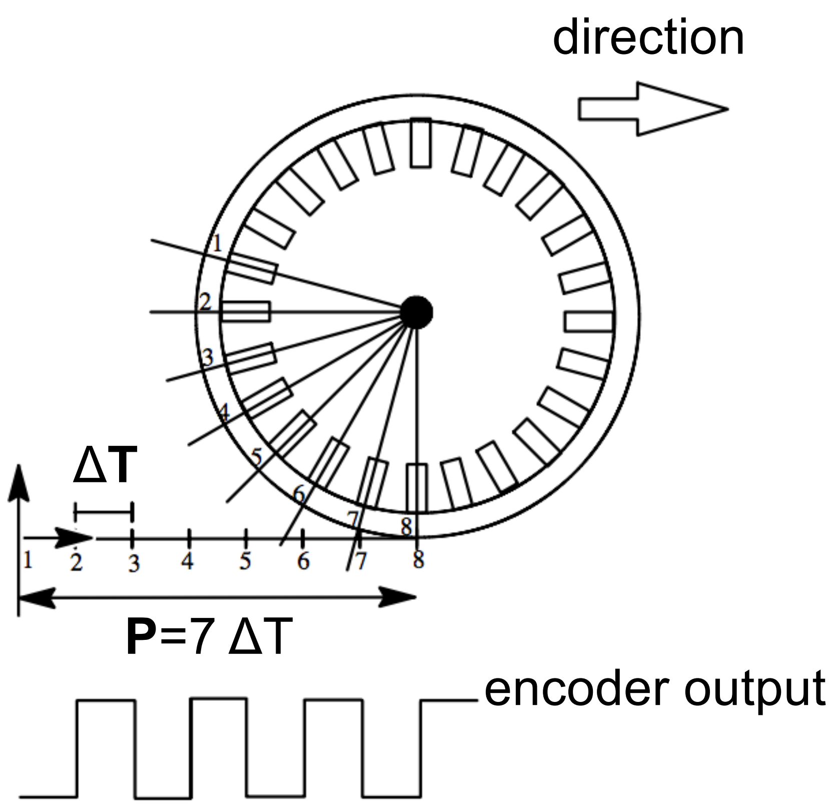
Introduction

- If you want a rover to navigate autonomously in a precise way, then the rover must know its position and orientation at any discrete time



Introduction

- The rover´s position **P** is obtained by integrating its translation **ΔT** over time
- **ΔT** is estimated from encoder readings of how much the wheels turned (wheel odometry)



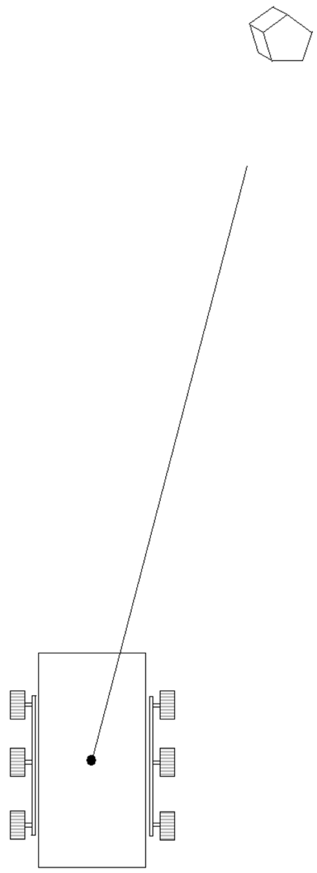
Introduction

- Wheel odometry fails on slippery environments, such as sandy terrain, due to the loss of traction

Courtesy NASA/JPL-Caltech

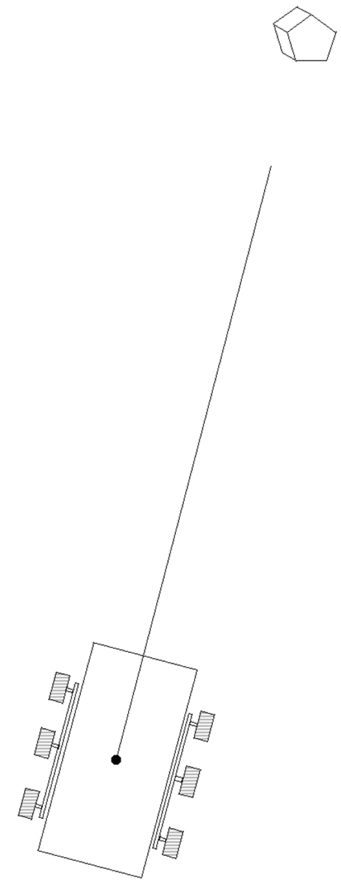
Introduction

- This could cause the rover to deviate from its desired path



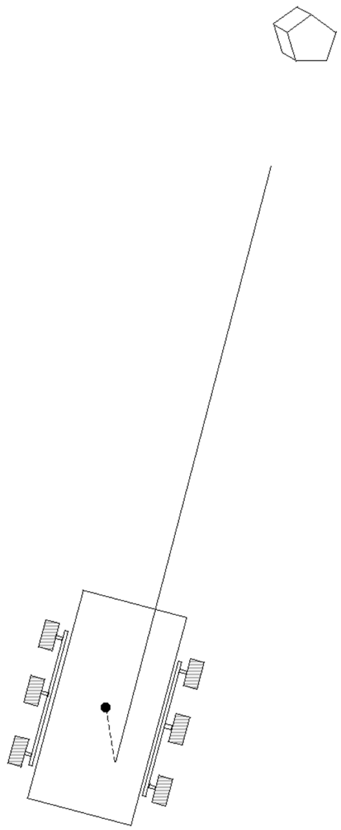
Introduction

- This could cause the rover to deviate from its desired path



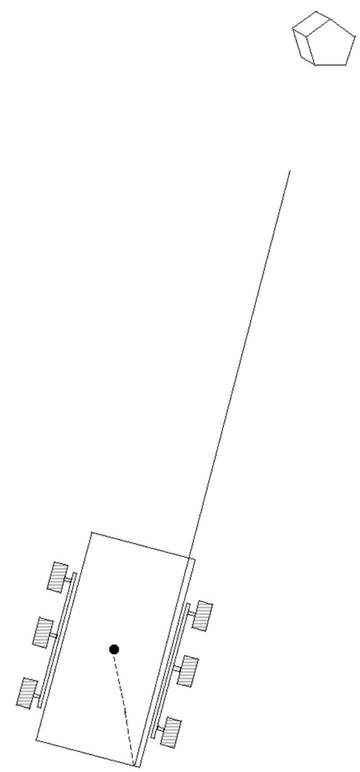
Introduction

- This could cause the rover to deviate from its desired path



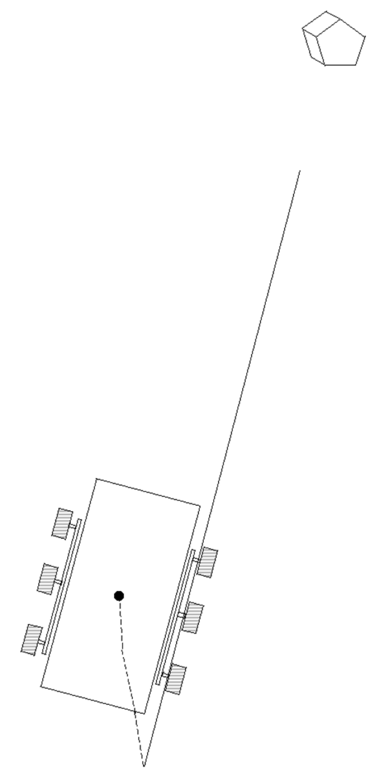
Introduction

- This could cause the rover to deviate from its desired path



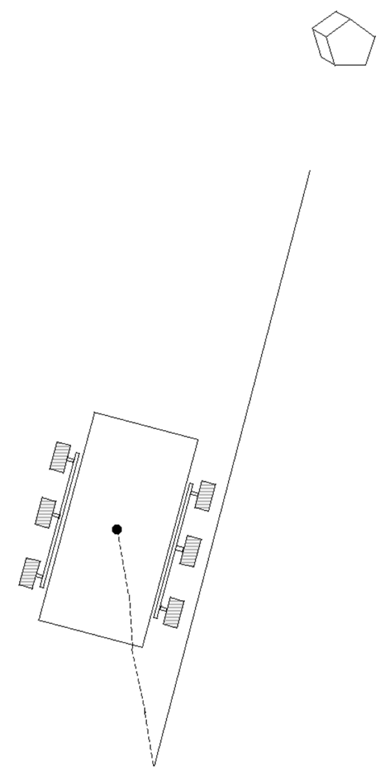
Introduction

- This could cause the rover to deviate from its desired path



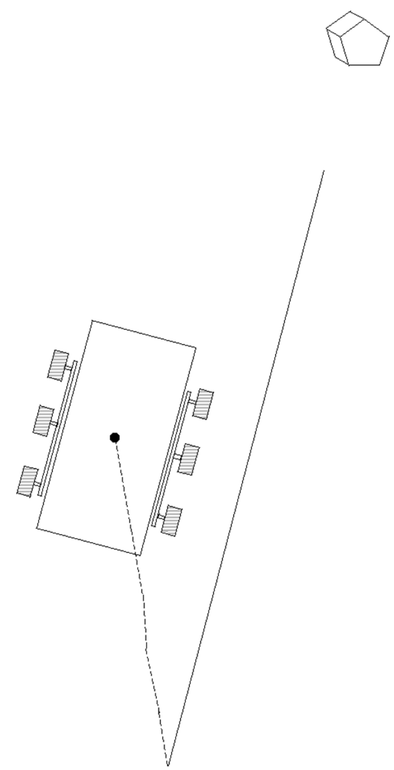
Introduction

- This could cause the rover to deviate from its desired path



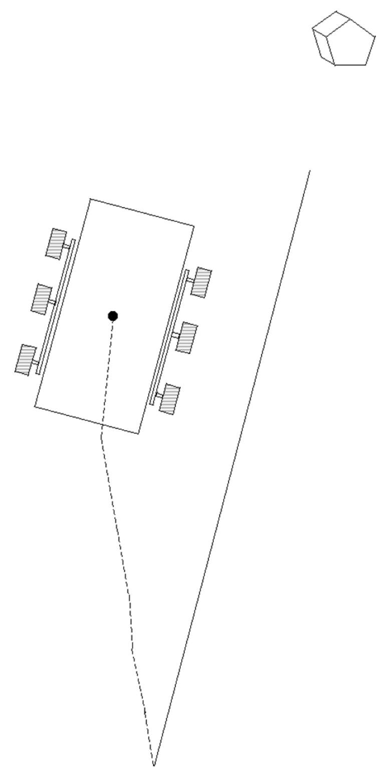
Introduction

- This could cause the rover to deviate from its desired path



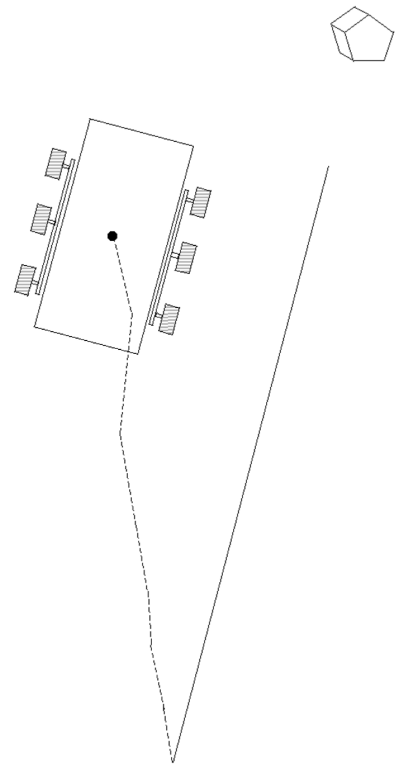
Introduction

- This could cause the rover to deviate from its desired path



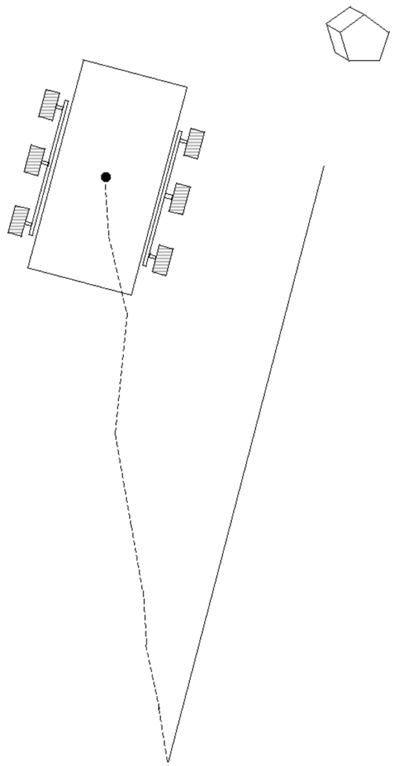
Introduction

- This could cause the rover to deviate from its desired path



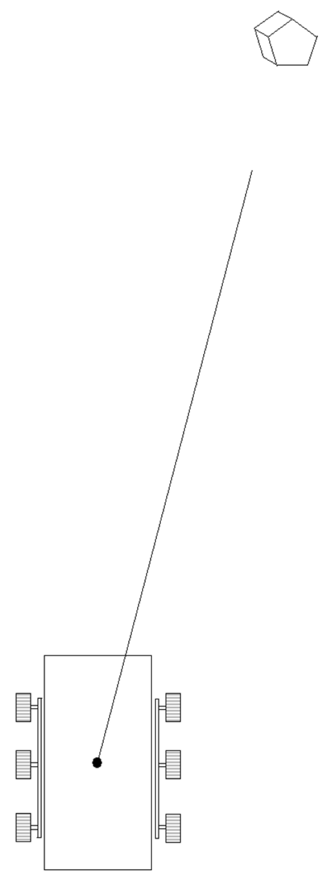
Introduction

- This could cause the rover to deviate from its desired path



Introduction

The problem is solved by detecting and compensating any slip that may occur

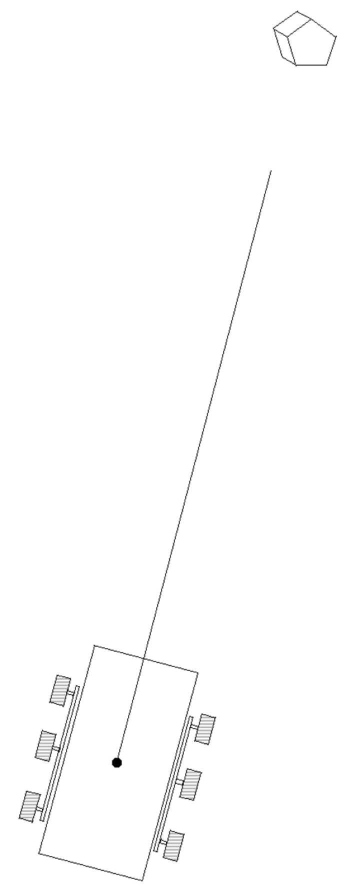


- Compute the robot 3D pose by minimizing the reprojection error at feature points
  - a stereo camera used
  - feature point correspondences required
- Knowing the rover 3D pose, detect and compensate any slip that may occur

Introduction

The problem is solved by detecting and compensating any slip that may occur

- Compute the robot 3D pose by minimizing the reprojection error at feature points
  - a stereo camera used
  - feature point correspondences required
- Knowing the rover 3D pose, detect and compensate any slip that may occur

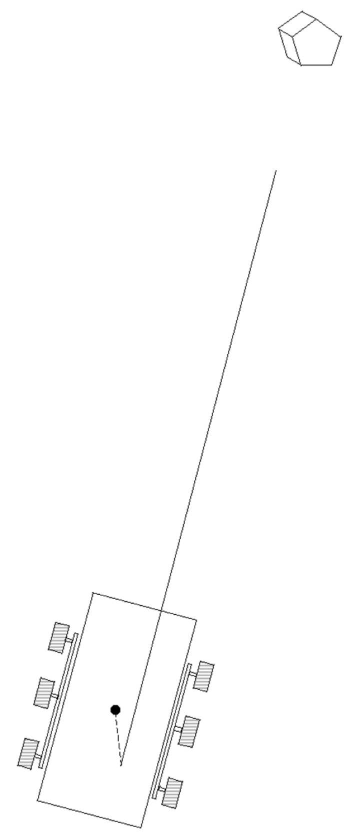


Introduction

The problem is solved by detecting and compensating any slip that may occur

- Compute the robot 3D pose by minimizing the reprojection error at feature points
  - a stereo camera used
  - feature point correspondences required

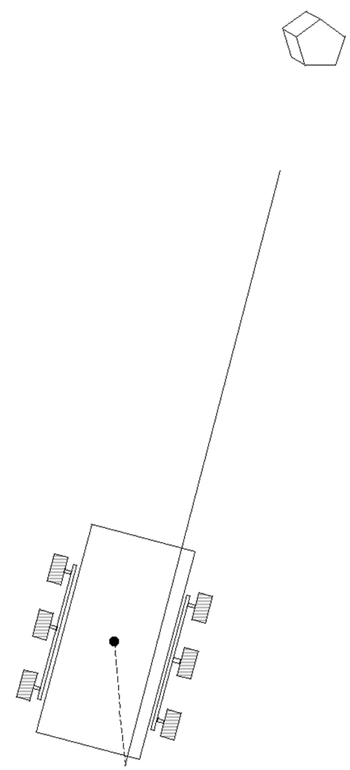
- Knowing the rover 3D pose, detect and compensate any slip that may occur



Introduction

The problem is solved by detecting and compensating any slip that may occur

- Compute the robot 3D pose by minimizing the reprojection error at feature points
  - a stereo camera used
  - feature point correspondences required
- Knowing the rover 3D pose, detect and compensate any slip that may occur

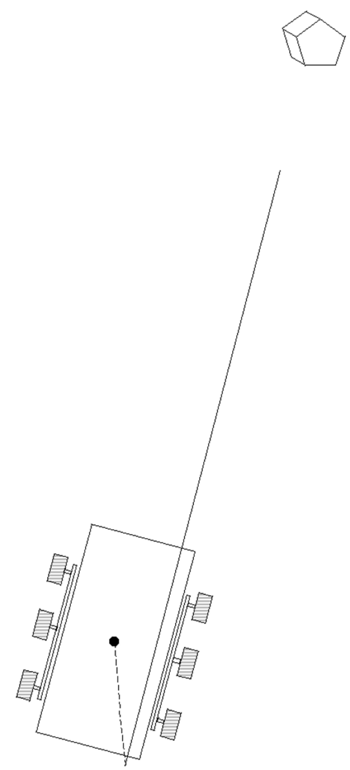


Introduction

The problem is solved by detecting and compensating any slip that may occur

- Compute the robot 3D pose by minimizing the reprojection error at feature points
  - a stereo camera used
  - feature point correspondences required

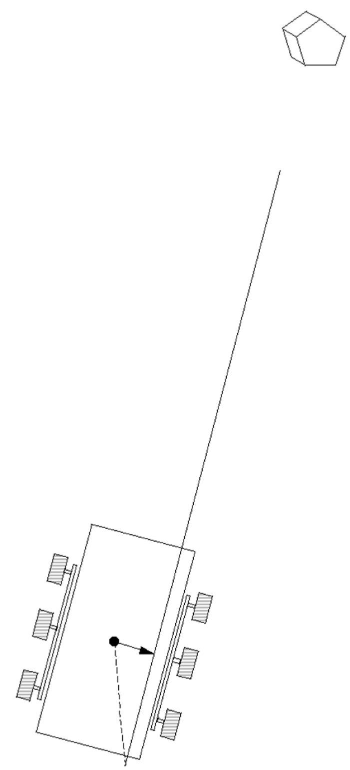
- Knowing the rover 3D pose, detect and compensate any slip that may occur



Introduction

The problem is solved by detecting and compensating any slip that may occur

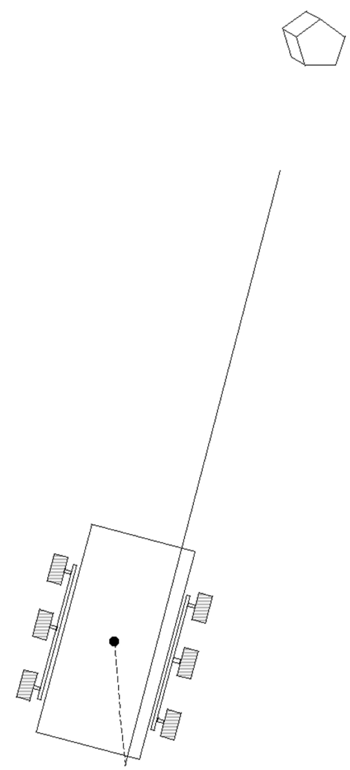
- Compute the robot 3D pose by minimizing the reprojection error at feature points
  - a stereo camera used
  - feature point correspondences required
- Knowing the rover 3D pose, detect and compensate any slip that may occur



Introduction

The problem is solved by detecting and compensating any slip that may occur

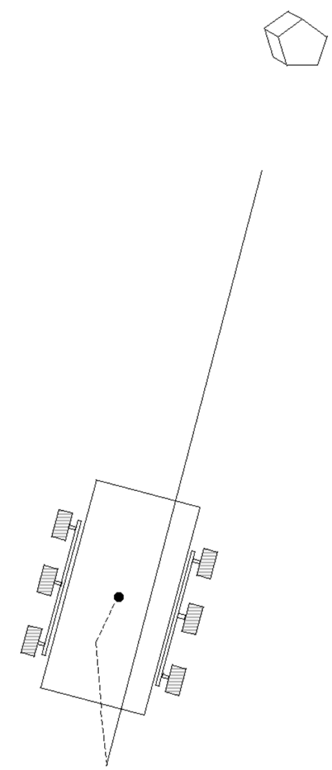
- Compute the robot 3D pose by minimizing the reprojection error at feature points
  - a stereo camera used
  - feature point correspondences required
- Knowing the rover 3D pose, detect and compensate any slip that may occur



Introduction

The problem is solved by detecting and compensating any slip that may occur

- Compute the robot 3D pose by minimizing the reprojection error at feature points
  - a stereo camera used
  - feature point correspondences required

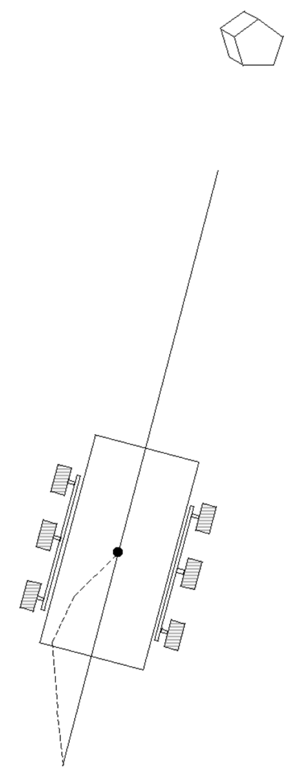


- Knowing the rover 3D pose, detect and compensate any slip that may occur

Introduction

The problem is solved by detecting and compensating any slip that may occur

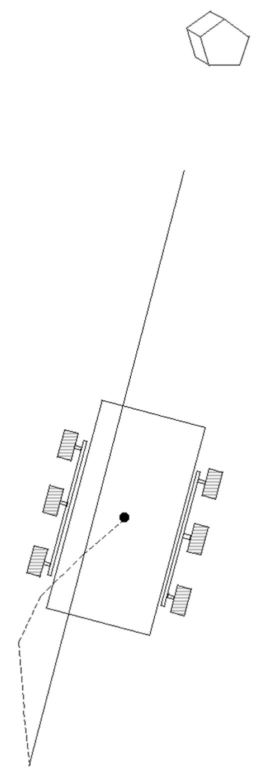
- Compute the robot 3D pose by minimizing the reprojection error at feature points
  - a stereo camera used
  - feature point correspondences required
- Knowing the rover 3D pose, detect and compensate any slip that may occur



Introduction

The problem is solved by detecting and compensating any slip that may occur

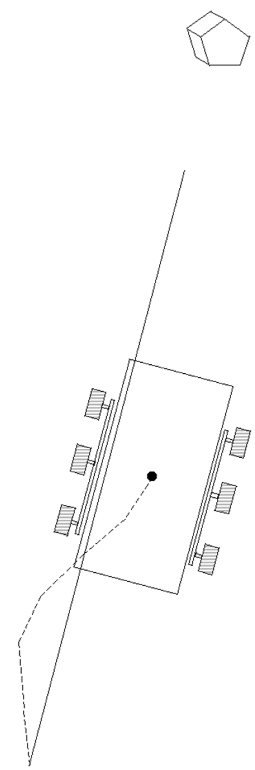
- Compute the robot 3D pose by minimizing the reprojection error at feature points
  - a stereo camera used
  - feature point correspondences required
- Knowing the rover 3D pose, detect and compensate any slip that may occur



Introduction

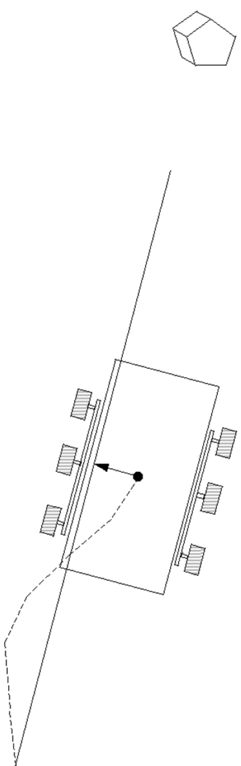
The problem is solved by detecting and compensating any slip that may occur

- Compute the robot 3D pose by minimizing the reprojection error at feature points
  - a stereo camera used
  - feature point correspondences required
- Knowing the rover 3D pose, detect and compensate any slip that may occur



Introduction

The problem is solved by detecting and compensating any slip that may occur

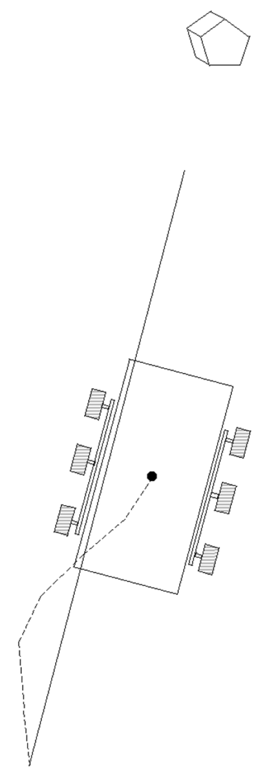


- Compute the robot 3D pose by minimizing the reprojection error at feature points
  - a stereo camera used
  - feature point correspondences required
- Knowing the rover 3D pose, detect and compensate any slip that may occur

Introduction

The problem is solved by detecting and compensating any slip that may occur

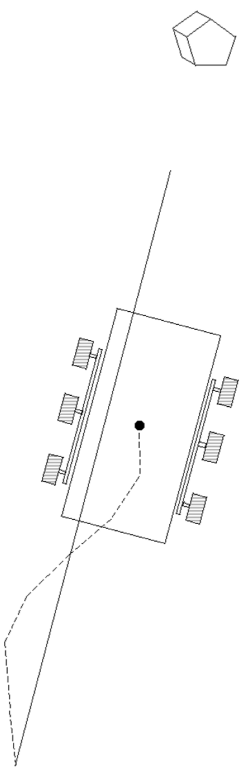
- Compute the robot 3D pose by minimizing the reprojection error at feature points
  - a stereo camera used
  - feature point correspondences required
- Knowing the rover 3D pose, detect and compensate any slip that may occur



Introduction

The problem is solved by detecting and compensating any slip that may occur

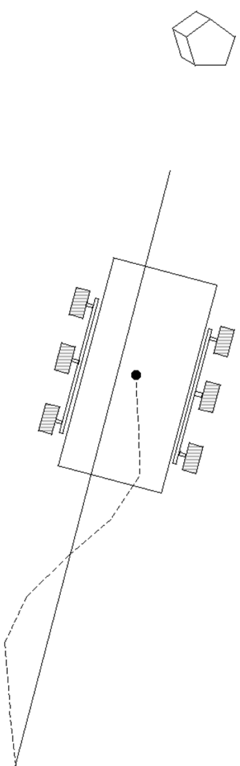
- Compute the robot 3D pose by minimizing the reprojection error at feature points
  - a stereo camera used
  - feature point correspondences required
- Knowing the rover 3D pose, detect and compensate any slip that may occur



Introduction

The problem is solved by detecting and compensating any slip that may occur

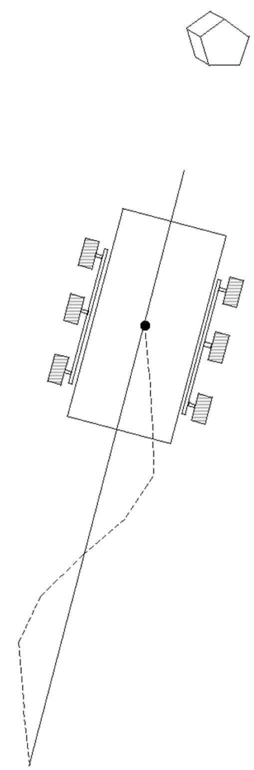
- Compute the robot 3D pose by minimizing the reprojection error at feature points
  - a stereo camera used
  - feature point correspondences required
- Knowing the rover 3D pose, detect and compensate any slip that may occur



Introduction

The problem is solved by detecting and compensating any slip that may occur

- Compute the robot 3D pose by minimizing the reprojection error at feature points
  - a stereo camera used
  - feature point correspondences required
- Knowing the rover 3D pose, detect and compensate any slip that may occur



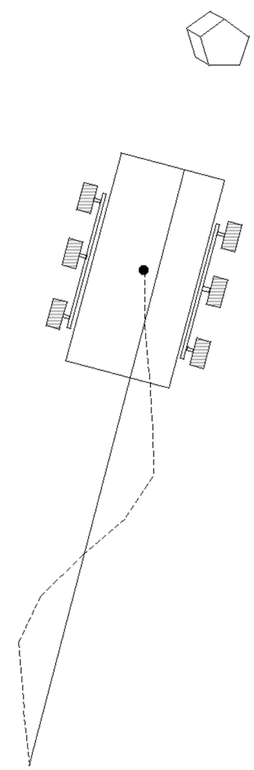
UNIVERSIDAD DE COSTA RICA

IPCV-LAB robot VISION Universidad de Costa Rica

Introduction

The problem is solved by detecting and compensating any slip that may occur

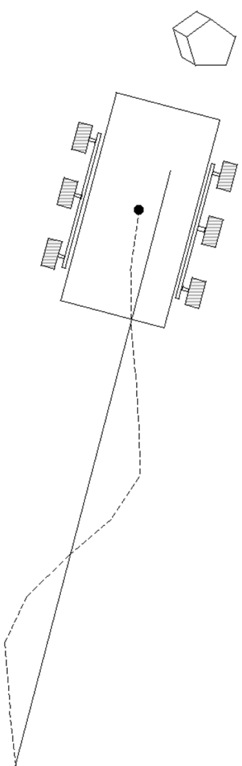
- Compute the robot 3D pose by minimizing the reprojection error at feature points
  - a stereo camera used
  - feature point correspondences required
- Knowing the rover 3D pose, detect and compensate any slip that may occur



Introduction

The problem is solved by detecting and compensating any slip that may occur

- Compute the robot 3D pose by minimizing the reprojection error at feature points
  - a stereo camera used
  - feature point correspondences required
- Knowing the rover 3D pose, detect and compensate any slip that may occur

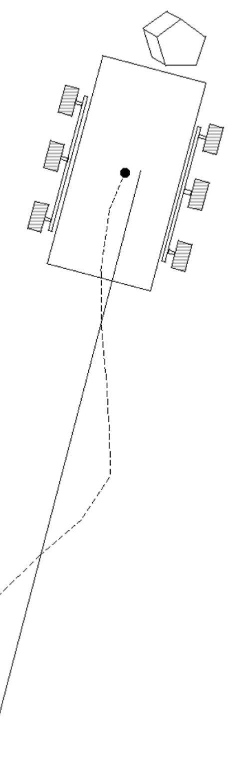


Introduction

The problem is solved by detecting and compensating any slip that may occur

- Compute the robot 3D pose by minimizing the reprojection error at feature points
  - a stereo camera used
  - feature point correspondences required

- Knowing the rover 3D pose, detect and compensate any slip that may occur

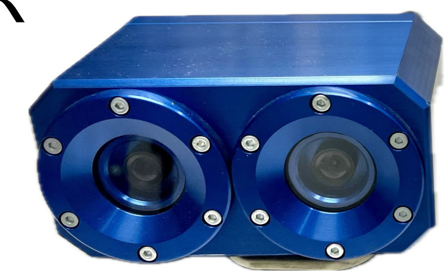


Approach

Sensor: instead of using a stereo camera, we propose to use a NIR ToF camera

- Monocular
- Faster depth signal calculation
- More accurate depth signal
- Easier to calibrate
- Works also at night
- Smaller
- Lower energy comsumption (?)

X



Stereo camera

✅

NIR ToF camera

Approach

Algorithm: instead of minimization the reprojection error at feature points, we propose the minimization of the photometric error at observation points

- no correspondences required
- faster

Two steps for getting the robot 3D pose:

(1) Frame to frame robot 3D motion estimation
(2) Integration of the estimated frame to frame robot 3D motion over time

Algorithm

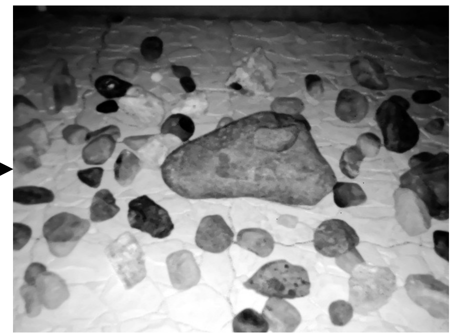
Basler
640 20gm_850nm
NIR ToF camera

IPCV-LAB robot VISION Universidad de Costa Rica

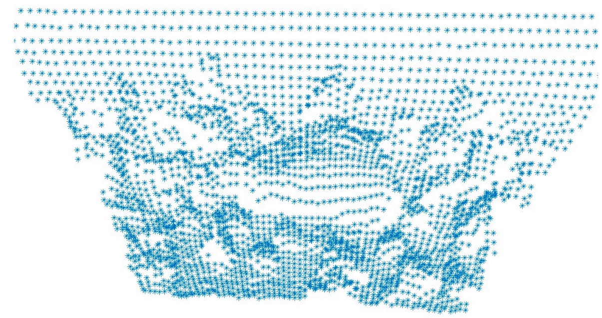
Algorithm

Basler
640 20gm_850nm
NIR ToF camera

NIR video signal (VGA 15 fps)



ToF depth signal (15 pcps)

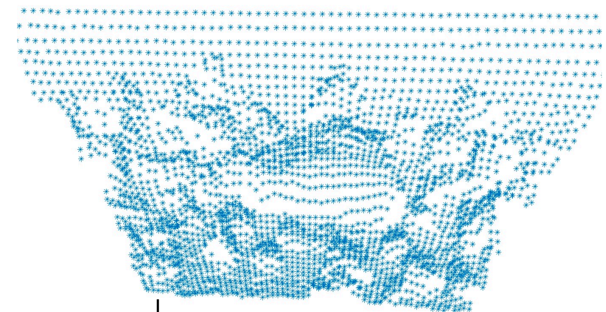


Algorithm

Basler
640 20gm_850nm
NIR ToF camera

NIR video signal (VGA 15 fps)

ToF depth signal (15 pcps)



Frame to frame robot 3D motion estimation

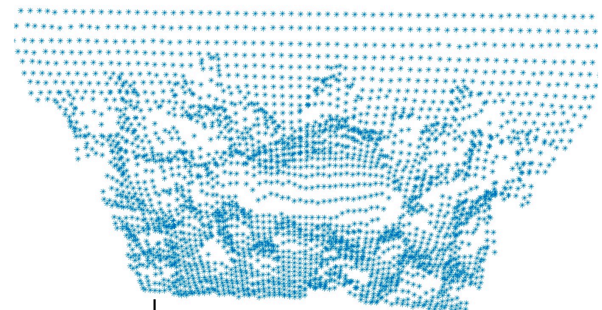
Robot 3D motion

Algorithm

Basler
640 20gm_850nm
NIR ToF camera

NIR video signal (VGA 15 fps)

ToF depth signal (15 pcps)



Frame to frame robot 3D motion estimation

Minimizing photometric error at observation points (iterative Maximim-Likelihood estimator)

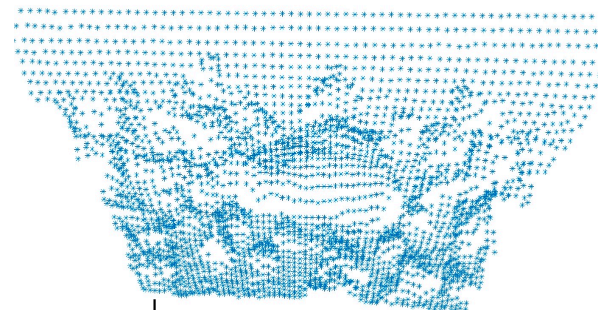
Robot 3D motion

Algorithm

Basler
640 20gm_850nm
NIR ToF camera

NIR video signal (VGA 15 fps)

ToF depth signal (15 pcps)



No correspondences required

Frame to frame robot 3D motion estimation

Minimizing photometric error at observation points (iterative Maximim-Likelihood estimator)

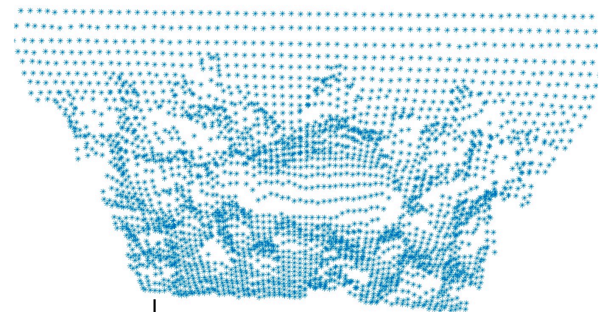
Robot 3D motion

Algorithm

Basler
640 20gm_850nm
NIR ToF camera

NIR video signal (VGA 15 fps)

ToF depth signal (15 pcps)



No correspondences required

Frame to frame robot 3D motion estimation

Minimizing photometric error at observation points (iterative Maximim-Likelihood estimator)

Composition rules Σ

Robot 3D motion

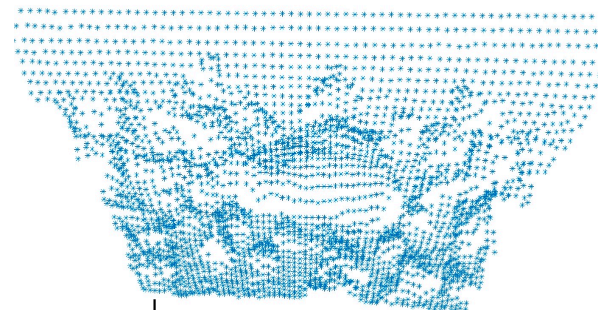
Robot 3D Pose

Algorithm

Basler
640 20gm_850nm
NIR ToF camera

NIR video signal (VGA 15 fps)

ToF depth signal (15 pcps)



No inertial or wheel odometry aid used so far

No correspondences required

Frame to frame robot 3D motion estimation

Minimizing photometric error at observation points (iterative Maximim-Likelihood estimator)

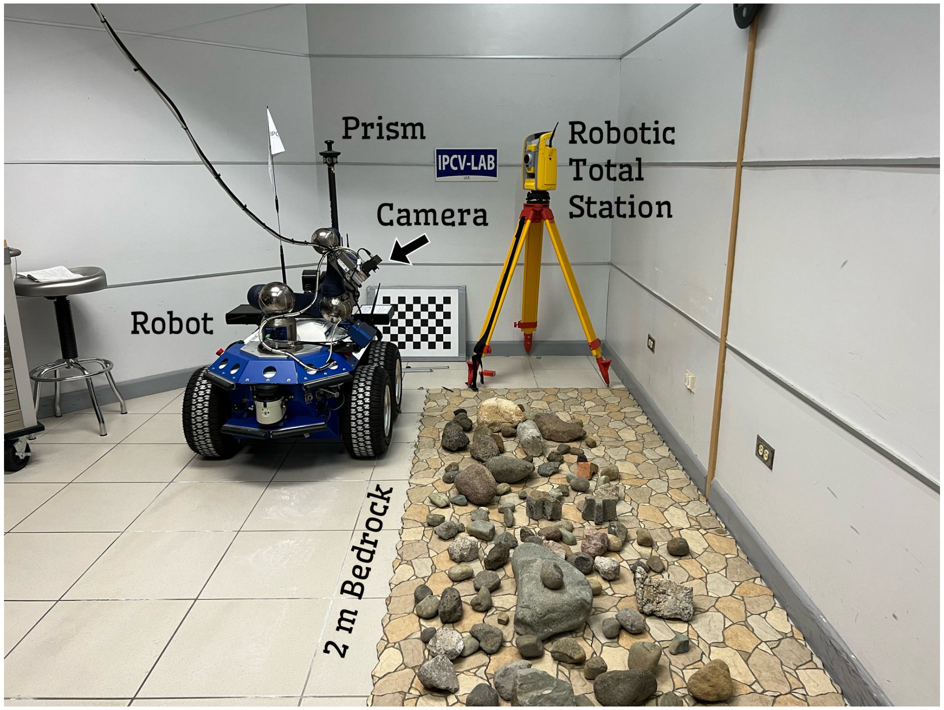
Composition rules Σ

Robot 3D motion

Robot 3D Pose

Experimental Results

- Implemented in C
  - under Ubuntu/ROS
- Tested in real rover
  - Seekur Jr.
- NIR ToF Camera
  - 80 cm above the ground
  - Looking to the left side of the rover tilted downwards 37°
- Ground truth
  - Robotic total station
- 180 over rough terrain



Experiment setup.

Experimental Results

- **First set**
  - Straight paths
  - 0.5, 1.0, 1.5 and 2.0 m
  - 3 cm/s
- **Mean of the absolute position error**
  - 0.47% ±0.32% of the distance traveled

- **Second set**
  - rotation on its own axis
  - 0.0° to 20.0°, 0.0° to 45.0° and 0.0° to 60.0°
  - 1.0°/s
- **Mean of the absolute orientation error**
  - 0.43% ±0.34% of the angle traveled

- **Real time operation**
  - 50 ±7 fps

Intel Core i9-9980HK CPU @ 2.40GHz and
8GB Memory under 64bit Ubuntu 14.04 LTS.

Summary

- NIR ToF Odometry algorithm proposed
  - Robot 3D pose computed by integrating frame-to-frame robot 3D motion over time
  - Robot 3D motion estimated by minimizing photometric error at observation points
  - No correspondences required
- Less 1% of the distance and angle traveled
- Real time operation
- Operation in the dark possible
- No inertial or wheel odometry aid used so far

Future work

- Improve robustness
- Adapt to fast moving robots
- Extent to SLAM
- More experiments in the dark