Autonomous Vehicle Simulation (AVS) Laboratory, University of Colorado

Basilisk Technical Memorandum

Document ID: Basilisk-houghCirlces

MODULE TO FIND CIRCLES IN AN IMAGE

Prepared by	T. Teil

Status: 1st Draft
Scope/Contents
This module either reads in a message of type `CameraImageMsg`, or a filename for an image. It then uses the OpenCV HoughCircles method to extract circles. It then writes these messages in a message of type `CirclesOpNavMsg`.

Rev	Change Description	By	Date
1.0	First release	T. Teil	May 24, 2019

Contents

---

1 Model Description

This module imports OpenCV which is installed if the cmake option is triggered. Two cases are possible: a filename for an image can be given, or an image message containing a size and pointer to image data can be used. If the message is read, the size is used to decode the image into an OpenCV matrix for image processing.

The following methods are then applied to the image:

- Greyscale
- Blurr
- Thresholding
- HoughCircles (which contains a Canny edge detection transform)

This allows for circles to be detected. The default parameters are set to be efficient at planet finding, as the text examples show.

2 Module Functions

- **Update State**: The image processing takes place in the Update method.

3 OpenCV Functions

More documentation is available on at https://docs.opencv.org/4.0.0/.

- void **cv::HoughCircles**(*InputArray* image, *OutputArray* circles, *int* method, *double* dp, *double* minDist, *double* param1 = 100, *double param2* = 100, *int* minRadius = 0, *int* maxRadius = 0)
- void **cv::cvtColor**(*nputArray* src, *OutputArray* dst, *int* code)
- double **cv::threshold**(*InputArray* src,*OutputArray* dst,*double* thresh,*double* maxval,*int* type)

- void **cv::blur**(*InputArray* src,*OutputArray* dst,*Size* ksize,*Point* anchor = Point(-1,-1),*int* borderType = BORDER_DEFAULT)

4 Module Assumptions and Limitations

The limitations of this module are in the image processing capabilities of the components. Current the main limitation is the lack of uncertainty measure around the circle estimates.

5 Test Description and Success Criteria

In order to test the proper function of this module, two test images are provided. The algorithm needs to find all the circles in the images within 1 pixel of relative error.

6 Test Parameters

Table 2: Error tolerance for each test.

Test image	Expected Circles	**Tolerated Error**
mars	1	1 px
moons	6	1 px

7 Test Results

The following table shows the results of the unit test described above.

Table 3: Test results

Check	**Pass/Fail**
mars	PASS
moons	PASS

The test does not generate the result image unless called explicitly from python in order to not add images to the repository.

8 User Guide

This section contains information directed specifically to users. It contains clear descriptions of what inputs are needed and what effect they have. It should also help the user be able to use the model for the first time.

- Construct algorithm and associated C++ container:
```
moduleConfig = houghCircles.HoughCircles()
moduleConfig.ModelTag = "houghCircles"
```
- Add test module to runtime call list:
```
unitTestSim.AddModelToTask(unitTaskName, moduleConfig)
moduleConfig.imageInMsgName = "sample_image"
moduleConfig.opnavCirclesOutMsgName = "circles"
```
- Image processing parameters:
```
moduleConfig.filename = imagePath
moduleConfig.expectedCircles = maxCircles
```

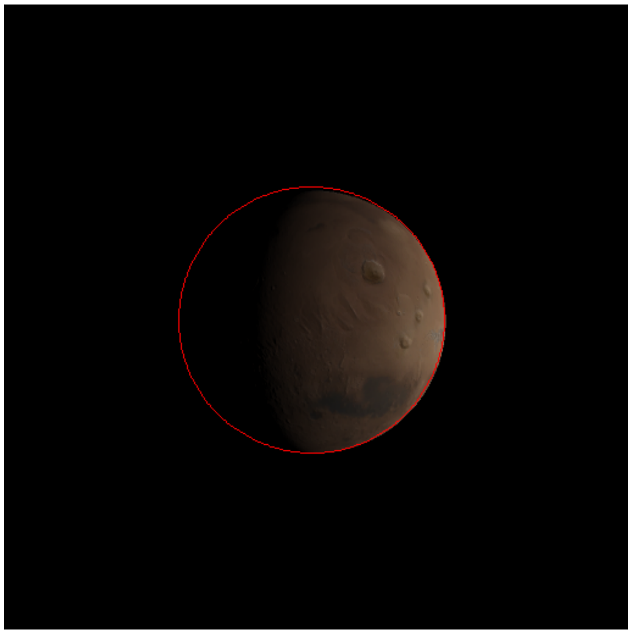
Fig. 1: Mars Circles

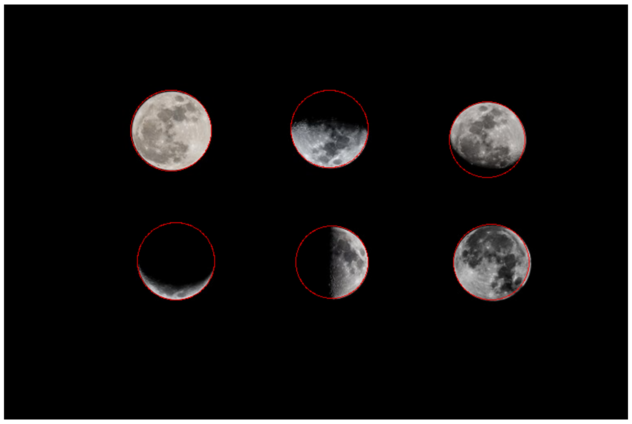
Fig. 2: Moon crescents circles

```
moduleConfig.cannyThresh1 = cannyHigh
moduleConfig.cannyThresh2 = cannyLow
moduleConfig.houghMinDist = minDist
moduleConfig.houghMinRadius = minRad
moduleConfig.blurrSize = blur
moduleConfig.dpValue = dp
moduleConfig.houghMaxRadius = 0
```