Analysis of Keypoint Detection Algorithm for Space-Based SFM

ADAM KING
Small Satellite Research Laboratory, University of Georgia
JUGAL PANCHAL
THINC Lab, University of Georgia

Abstract:

Structure from Motion (SFM), though a remarkably easy task for humans, is difficult for machines. The input to SFM is a set of features/keypoints of a set of images. Each keypoint must be sufficiently described and corresponded between at least two images. Herein, detailed analysis is performed to compare the SIFT, SURF, and ORB feature detectors against each other, as well as the Harris and Shi-Tomasi corner detectors against each other. The results of this testing will be used for development of the Small Satellite Research Laboratory (SSRL)'s Mapping and Ocean Color Imager (MOCI) Satellite mission, which requires research and development of custom feature detection, description, matching, and SFM software. Currently, such processes are insufficient for spaced-based imagery, and the custom software produced by the SSRL will develop towards landscape-scale 3D reconstruction. The results of this paper will help the SSRL to determine the best path forward with developing such algorithms. SIFT, ORB, and SURF are compared at the feature detection stage using average keypoint quantities at various parameters and keypoint computation time for parameter extrema, which is important for applications limited by power constraints. The same comparison is made between the Harris and Shi-Tomasi corner detectors. Visual inspection is also used to determine the quality of keypoints produced by each algorithm. Image data was produced by the Small Satellite Research Laboratory using Blender. Results indicate that ORB is superior to both SIFT and SURF and that the Shi-Tomasi corner detector is superior to the Harris corner detector.

Keywords: structure, motion, feature, keypoint, detect, SIFT, ORB, SURF, Harris, Shi-Tomasi, corner, small, satellite, SSRL, MOCI, AFRL, UGA

Introduction

Though humans (and all creatures with stereoscopic vision) can easily understand the world in three dimensions with only two-dimensional input, such a task is difficult for machines. There have been many strides in this field over the past few decades, as determining the 3D structure of object using 2D images enables machines to achieve perception (low-level signals or stimuli translated into high-level symbols) and cognition (high-level symbols related to real-world concept). These enable machines to reason (forming relationship between real-world concepts) and action (bringing about change in the state of the environment). One approach to creating 3D representations of the world from 2D models is the calculation of 3D geometry from 2D geometry, or Structure from Motion (SFM) [5].

There are several assumptions that are made to formulate the SFM task, as Jebara [5] breaks

down: Given a camera and a scene, it must be that one of these is moving and the other is static in position, and that at least two images/frames are gathered by the camera. The input to the SFM task is the set of 2D features of a scene, as processed beforehand by some module that has processed the images to consistently extract, locate, and describe these features. Features can include corners, edges, or whatever an algorithm deems to be an "important" point. These features are calculated using two or more of the images captured by the camera such that each feature in any given frame can be matched to a feature in at least one other frame, an outcome generally described as correspondence. This process can be noisy, error-prone, and computationally timely, with small changes greatly affecting the outcome of the SFM task. Detection of such features will be the focus herein by 1) considering the average number of keypoints produced by each algorithm and comparing these numbers across similar parameters, and 2) comparing the computation time of each algorithm at the extrema of the parameters. Detecting and matching feature points in itself is a "decidedly difficult computer vision problem which cannot be dismissed" [1].

Feature detection can be defined as follow [8]: feature detection is done after the pre-processing stage. The pre-processing stage includes image processing techniques such as binarization, thresholding, edge detection, etc. Detection becomes an important stage as relevant information, such as the shape of the objects, are extracted from the image. Next, features are whittled down by comparing each keypoint in an image to its nearest neighbor. If these keypoints are similar enough, either they will be merged, or one keypoint will be discarded.

The Small Satellite Research Laboratory (SSRL)'s research and development of the Mapping and Ocean Color Imager (MOCI) satellite is funded by the Air Force Research Laboratory (AFRL). This mission will [4] gather several images of pre-determined points of interest on Earth from varying angles in one passover, a method called "point tracking." These images will then be used to construct 3D representations of Earth. The research herein focuses on feature extraction, as working to minimize the runtime of the feature extraction algorithm will better the power limitations of the satellite, as less runtime means more power available to run the satellite.

This research will focus on testing the average number of keypoints with regards to similar parameters in SIFT, ORB, and SURF feature detectors as well as the Harris and Shi-Tomasi corner detectors. Runtime is compared in a similar fashion, with comparing average runtimes at parameter extrema. Visual inspection is also done to compare the quality of keypoints, as numerous keypoints not around objects of interest are of no use in feature matching. These comparisons were done successfully, and results indicate that ORB feature detector is superior to the SIFT and SURF feature detectors, while the Shi-Tomasi corner detector is indicated to be superior to the Harris corner detector.

Literature Review

SIFT: The Scale-Invariant Feature Transform algorithm proposed by Lowe [6] consists of four major steps: scale-space extrema detection, keypoint localization, contrast thresholding, and curvature thresholding, which can be described as follows [6]: the first stage in detection of scale-space extrema is the identification of locations and scales that "can be repeatedly be assigned under differing views of the same object," which is done by searching for features that are stable across all possible scales, namely by using the continuous scale function presented by Witkin, called "scale space" [6]. This is done by taking difference of adjacent Gaussian blur images/intervals, which is shown in Lowe's paper to be an approximation to the Laplacian of Gaussian function. The next step is keypoint localization, which performs a detailed fit to the surrounding pixels and intervals using a second-order Taylor expansion to interpolate the true location of an extrema to sub-pixel and sub-interval accuracy, following the extrema as necessary. This final extrema location is then used to calculate the contrast of an extrema when compared to the surrounding pixels and intervals, rejecting extrema whose contrast is

less than 3% the maximum change in intensity. The last stage calculates the ratio of principal curvatures by comparing the 2D Harris matrix eigenvalues to a ratio of change in contrast across an edge vs along an edge. The output of this is keypoints that are invariant to image scale and rotation, and are also robust against affine distortion, change in 3D viewpoint, addition of noise, and change in illumination.

ORB: The Oriented FAST and Rotated Brief algorithm's feature detection uses the FAST corner detector algorithm with a twist [7]. The FAST algorithm compares the intensity of a given pixel to the intensities of the surrounding pixels at a given radius around it, which in this case is 9. If the pixel's intensity value is either greater than all of the surrounding pixels or less than all of the surrounding pixels, the pixel is considered to be a corner. After these keypoints have been found, the Harris corner measure is applied to these, and the top N keypoints are selected for further processing. This process is made multi-scale by being run on a scale pyramid of an image. The intensity centroid is then used to determine the orientation of a corner. The intensity centroid makes the assumption that a corner's intensity is offset from its center. This offset is used to assign each keypoint an orientation, thus creating an "oriented" FAST algorithm. These keypoints are rotation invariant and resistant to noise. As this algorithm is compared to SIFT and SURF, it can be reasonably assumed these keypoints are also somewhat robust against addition of noise, change in illumination, change in 3D viewpoint, and affine distortion.

SURF: The Speeded-Up Robust Features algorithm begins with feature detection by using calculating the integral image before using a Hessian matrix-based feature detector [9]. A box filter defined by a 2D Hessian matrix is used to approximate the Laplacian of Gaussian. The box filter filter is convolved over the integral image to produce keypoints more quickly than if convolved over a non-integral image. This process is made multi-scale by being run on a scale pyramid of an image. These keypoints are scale- and rotation-invariant, and since this algorithm is compared to SIFT, it can be reasonably assumed that these keypoints are also somewhat robust against change in 3D viewpoint, change in illumination, affine distortion, and addition of noise.

Harris: The Harris Corner Detector uses a window of pixels and calculates the difference between the original and the moved window, the concept similar to auto-correlation [12]. The keypoints are determined in a window that produces large variation when moved in any direction [13]. Then getting the Sum of Squares Differences with the use of the Difference of Gaussian, equation represented in the matrix format, leads to calculation difference in x direction, y direction and the x-y direction. Generating this matrix also leaves with eigenvalues, which are then used to plot the graph of where the point lies with respect to the eigenvalues on the graph. This is where the threshold is used to filter out pixels and give only the most important corners.

Shi-Tomasi: Like the Harris Corner Detector, the Shi-Tomasi proceeds with calculating the eigenvalues from the matrix generated. The Shi-Tomasi detector uses a different approach to determine which corners are important by estimating the threshold of the vector by simply multiplying the two eigenvalues together [14].

Problem Description

For the MOCI mission, the on-board process will take the images and perform feature detection, description, and matching before putting this output through an SFM process [1]. This SFM process produces "sparse point clouds," which are the matched keypoints projected onto a 3D field using geometry [1]. This point cloud is then put into CVMS/PVMS processes to produce first a "dense point clouds,", which is the sparse point clouds overlayed with image snippets [1]. The process then produces a "mesh,", which is a continuous surface approximation of the 3D field based on the dense point cloud [1]. Downlinking only these meshes is the end goal of the project, as this would mean quick,

undetectable 3D modeling of anywhere on Earth. This data could also be improved using higher resolution imaging devices [1].

The mission will perform the feature detection, description, and matching [4] using an FPGA, and will then use an NVIDIA Jetson TX1 GPU to run the SFM and CVMS/PVMS processes. Because the satellite is such a small form factor, a 10x10x33cm prism, there is limited room for batteries. The current design is for a 40Whr battery, but even with this, power limitations are a great concern. To combat this, research is being done into a custom feature extraction, SFM, and CVMS/PVMS process. Determining which algorithms perform the most efficiently with regards to runtime and keypoint quantity and quality will enable further research to be done with the custom process. In order to determine the most efficient algorithm, the runtimes, average keypoint quantities, and keypoint quality were compared.

The implementations of SIFT, ORB, SURF, Harris, and Shi-Tomasi detectors that were used are from the Python/C++ opensource library, OpenCV, version 3.2.0. As the SIFT and SURF algorithms are patented, they are found in the opencv_contrib-3.2.0 package, and the ORB, Harris, and Shi-Tomasi algorithms are found in the opencv-3.2.0 package. As this is an opensource library, most code is experimental, and therefore prone to possible errors, such as discussed later. The testing code was written in Python, and calls C++ implementations of SIFT, SURF, and ORB that were designed by the opensource community using the respective research papers as a basis. ORB is actually commissioned by the OpenCV project, and was developed specifically to be a restriction-free feature detection and description algorithm that is comparable to SIFT and SURF. These implementations, along with the Harris and Shi-Tomasi implementations from OpenCV, permitted access to a wide variety of tunable parameters for each algorithm, which allowed for exhaustive testing outside of the default values provided by either the implementations or the respective research papers. The project was integrated with Excel using the openpyxl library to enable easier data manipulation.

Test Images: Previously, the SSRL had produced a simulated data set for satellite images using Blender. The output was 40 images that track a point on the simulated Earth as the simulated satellite orbits around the point while point tracking from various angles. The objects in these images are of landscape-scale, such as mountains, but are also regular shapes to allow for later determining how to lessen the noise produced by the SFM/CVMS/PVMS process.

Parameter Extrema: For this paper, first upper and lower bounds for each parameter for each algorithm were found. Some parameters were found not to affect the outcome on the number of keypoints, and thus were ignored for exhaustive testing purposes. Parameter extremas were determined by testing the parameters individually with all other parameters being set to default according to OpenCV's implementation. After a pattern was discovered for a given parameter, the extrema values were chosen where the number of keypoints produced by that particular iteration seemed stable, where the number of keypoints was something unusable, such as producing no keypoints, or where the number of keypoints output at a particular extrema did not vary much at the immediately surrounding values of the parameter. Step sizes were determined by comparing the number of keypoints output at different step sizes. Step sizes were chosen to be the minimum value that produced a slight, but not wild, difference between a run at a given parameter and a run at a parameter one step size away. After this, step sizes were also modified to improve runtime of the exhaustive testing scheme, namely trying to keep the number of times each parameter would change less than ~8 when possible, and less than ~16 otherwise. The following parameter extrema values are inclusive.

Timing: To compare SIFT, ORB, and SURF, as well as Harris and Shi-Tomasi, timing tests were performed with each algorithm using all possible extrema permutations; i.e., ORB was tested using the following parameter values:

(intervals=1, edgeT=0, fastT=1)
(intervals=1, edgeT=0, fastT=41)
(intervals=1, edgeT=240, fastT=1)
(intervals=1, edgeT=240, fastT=41)
(intervals=13, edgeT=0, fastT=1)
(intervals=13, edgeT=0, fastT=41)
(intervals=13, edgeT=240, fastT=1)
(intervals=13, edgeT=240, fastT=41)

Each of these permutations was tested using a timing function 3 times, producing an average runtime for the extreme parameter values for each algorithms. Though SIFT, ORB, and SURF were timed using the UNIX `time' function and Harris and Shi-Tomasi were timed using python's time.time() function, this difference is trivial since the comparison are not made between SIFT/ORB/SURF and Harris/Shi-Tomasi. Timing was also performed across spatial resolutions, namely 480p vs 1560p, both at 16:9 ratio. All timing tests were performed on the same image. The system time was chosen from the UNIX time function, which is inherently the output of the python time function. User time should not be measured here, as scheduling procedures vary wildly from computer to computer and OS to OS.

Keypoint Quantity: To test the keypoint quantity, an exhaustive program was made to calculate the average number of keypoints produced at a parameter level similar across algorithms. For example, to compare the keypoint quantities between SIFT, ORB, and SURF, the average number of keypoints produced by all permutations of the available parameters for each algorithm were calculated. To elaborate further, these three algorithms were compared with the intervals per octave parameter. To do this, these algorithms were run with the following scheme (example is for SIFT):

1. for each interval value in the range of <minimum intervals> to <maximum intervals> at step size 1
   1.1. temporary sum = 0
   1.2. temporary counter = 0
   1.3. for each curvature threshold in the range of <minimum curvature> to <maximum curvature> at step size 2
      1.3.1. for each contrast threshold in the range of <minimum contrast> to <maximum contrast>
         1.3.1.1. use the SIFT keypoint detector with the current parameter values
         1.3.1.2. if the number of keypoints produced is less than or equal to the maximum number of keypoints allowed and is greater than or equal to the minimum number of keypoints allowed
            1.3.1.2.1. temporary sum += number of keypoints at this iteration
            1.3.1.2.2. temporary counter ++
   1.4. after mitigating a divide by zero error, the average number of keypoints at the current interval value is <temporary sum> / <temporary counter>

A similar scheme was used for the parameters of ORB and SURF. Extending this comparison at the intervals level, the Harris and Shi-Tomasi corner detectors were compared at the contrast level in similar fashion. Such testing was done for every other image for a total of 20 images being tested in this manner. Original plans slated that this process also be tested at varying spatial resolutions as the

initial input, though an explanation to why this was later abandoned is discussed in later sections. The maximum and minimum number of keypoints threshold for SIFT, ORB, and SURF is to discard any outliers before calculating the average, and the same is true for the minimum and maximum number of corners for Harris and Shi-Tomasi. A maximum keypoint quantity of 3000 was chosen because it is 100 times greater than the minimum keypoint quantity of 30, which was chosen based on the assumption that few than 30 keypoints would not yield a high enough probability to produce good results for image correspondence. A maximum corner quantity of 80 was chosen because it is 10 times greater than the minimum corner quantity of 8, which was chosen based on counting the number of corners of objects of interest in any given image and determining the least number of most important corners.

Keypoint Quality: To test the quality of keypoints, a more heuristic was taken. Visually comparing all images across all parameters for all algorithms would be infeasable, as more than an order of magnitude greater than (13 intervals) * (20 images) * (3 algorithms) number of images would have been produced for just comparing SIFT, ORB, and SURF. With this, images were output with the keypoints corresponding to the permutations of parameter extremas, similar to how the timing tests were conducted. Also, an image was produced for each algorithm at the halfway point of each parameter concurrently. Finally, images were also produced using the default setting of OpenCV, if the algorithm allowed. Factors taken into account were the spread of keypoints as well as the closeness of keypoints to the simulated landscape-scale objects at the center of each image.

Results

Parameter Extrema: The following parameters were determined for each algorithm:

Table 1: SIFT Parameters

Parameter	Minimum	Maximum	Step Size
Intervals per octave	1	13	2
Edge (Curvature) Threshold	0	24	2
Contrast Threshold	0.02	0.10	0.02

Table 2: ORB Parameters

Parameter	Minimum	Maximum	Step Size
Intervals per octave	1	13	1
Edge Threshold	0	240	20
FAST Threshold	1	41	10

Table 3: SURF Parameters			
Parameter	**Minimum**	**Maximum**	**Step Size**
Intervals per octave	1	13	1
Hessian Threshold	295	505	3

Table 4: Harris Parameters			
Parameter	**Minimum**	**Maximum**	**Step Size**
Contrast Threshold	0.01	0.31	0.02
Sobel Aperature Size	1	9	2
Corner Measure Threshold	0.64	0.98	0.02

Table 5: Shi-Tomasi Parameters			
Parameter	**Minimum**	**Maximum**	**Step Size**
Contrast Threshold	0.01	0.31	0.02
Minimum Corner Separation	5	25	5
Corner Quality	0.01	0.09	0.01

Timing: The following output shows the result of the process described in the previous section:

Table 6: SIFT Timing

RESOLUTION	INTV / OCTV	EDGE	CONTRAST	TIME 1	TIME 2	TIME 3	AVG	AVG @INTV/OCTV
480	1	0	0.02	0.068	0.072	0.076	0.072	0.0843333333
			0.1	0.096	0.096	0.108	0.1	
		24	0.02	0.092	0.1	0.072	0.088	
			0.1	0.076	0.084	0.072	0.0773333333	
	13	0	0.02	0.216	0.152	0.18	0.1826666667	0.1796666667
			0.1	0.192	0.196	0.16	0.1826666667	
		24	0.02	0.168	0.144	0.168	0.16	
			0.1	0.224	0.152	0.204	0.1933333333	
1560	1	0	0.02	0.296	0.296	0.268	0.2866666667	0.304
			0.1	0.312	0.288	0.328	0.3093333333	
		24	0.02	0.312	0.32	0.32	0.3173333333	
			0.1	0.252	0.308	0.348	0.3026666667	
	13	0	0.02	1.644	0.952	1.084	1.2266666667	1.019
			0.1	0.896	0.864	0.916	0.892	
		24	0.02	0.888	0.936	0.944	0.9226666667	

			0.1	1.024	1.024	1.056	1.0346666667	

Table 7: ORB Timing 1

RESOLUTION	INTV / OCTV	EDGE	FAST	TIME 1	TIME 2	TIME 3	AVG	AVG @INTV/OCTV
480	1	0	1	0.044	0.04	0.064	0.0493333333	0.0486666667
			41	0.052	0.052	0.056	0.0533333333	
		240	1	0.08	0.036	0.024	0.0466666667	
			41	0.044	0.056	0.036	0.0453333333	
	13	0	1	0.076	0.048	0.056	0.06	0.0576666667
			41	0.048	0.052	0.076	0.0586666667	
		240	1	0.072	0.044	0.052	0.056	
			41	0.04	0.084	0.044	0.056	
1560	1	0	1	0.064	0.044	0.052	0.0533333333	0.06
			41	0.072	0.056	0.044	0.0573333333	
		240	1	0.088	0.064	0.072	0.0746666667	
			41	0.072	0.04	0.052	0.0546666667	
	13	0	1	0.064	0.064	0.06	0.0626666667	0.074
			41	0.084	0.068	0.084	0.0786666667	
		240	1	0.08	0.088	0.084	0.084	
			41	0.068	0.084	0.06	0.0706666667	

Table 8: ORB Timing 2

RESOLUTION	INTV / OCTV	EDGE	FAST	TIME 1	TIME 2	TIME 3	AVG	AVG @INTV/OCTV
480	1	0	1	0.044	0.048	0.048	0.0466666667	0.0443333333
			41	0.048	0.04	0.032	0.04	
		240	1	0.048	0.052	0.036	0.0453333333	
			41	0.04	0.048	0.048	0.0453333333	
	13	0	1	0.02	0.032	0.052	0.0346666667	0.0523333333
			41	0.048	0.052	0.032	0.044	
		240	1	0.056	0.076	0.092	0.0746666667	
			41	0.064	0.06	0.044	0.056	
1560	1	0	1	0.056	0.052	0.032	0.0466666667	0.044
			41	0.044	0.04	0.04	0.0413333333	
		240	1	0.04	0.048	0.032	0.04	
			41	0.036	0.052	0.056	0.048	
	13	0	1	0.088	0.128	0.104	0.1066666667	0.104
			41	0.076	0.08	0.064	0.0733333333	
		240	1	0.096	0.096	0.14	0.1106666667	
			41	0.108	0.112	0.156	0.1253333333	

Table 9: SURF Timing

RESOLUTION	INTV / OCTV	HESSIAN	TIME 1	TIME 2	TIME 3	AVG	AVG @INTV/OCTV
480	1	295	0.072	0.052	0.08	0.068	0.0573333333
		505	0.048	0.044	0.048	0.0466666667	
	13	295	0.088	0.092	0.092	0.0906666667	0.102
		505	0.104	0.108	0.128	0.1133333333	
1560	1	295	0.112	0.12	0.16	0.1306666667	0.14
		505	0.132	0.152	0.164	0.1493333333	
	13	295	0.316	0.344	0.388	0.3493333333	0.3586666667
		505	0.344	0.356	0.404	0.368	

SIFT, ORB, and SIFT are compared at the intervals level. As above, for an image at 480p with 1 interval per octave, ORB outperforms both SIFT and SURF, even finishing in nearly half the time of SIFT. The story is the same for an image at 480p with 13 intervals per octave, with ORB completing in half the time SURF does and a third the time SIFT does. When looking at the tests run with an image at 1560p, things begin to seem too good to be true, so it is assumed they are. After running a second timing test on ORB, a similar story persists. The data shows that when using ORB, and image at 480p with 1 interval per octave and an image at 1560p with 1 interval per octave take roughly the same amount of time when compared at the interval level. Also, a similar story occurs with 13 intervals per octave, with a 1560p image taking only twice as long as a 480p image to process. This seems very irrational, given that a 1560p image has ~10.5 times as many pixels as a 480p image. Discussion about this can be found in the following section. Discarding the ORB timing tests, it is clear that SURF outperforms SIFT on a purely speed basis.

Table 10: Harris Timing

RESOLUTION	CONTRAST	SOBEL	TIME 1	TIME 2	TIME 3	AVERAGE	AVG @ CONSTRAST
480	0.01	1	0.038	0.0395	0.0405	0.0393333333	0.0454166667
		9	0.052	0.049	0.0535	0.0515	
	0.3	1	0.0475	0.049	0.0475	0.048	0.0515833333
		9	0.056	0.059	0.0505	0.0551666667	
1560	0.01	1	0.043	0.04	0.034	0.039	0.0463333333
		9	0.06	0.049	0.052	0.0536666667	
	0.3	1	0.0495	0.0485	0.045	0.0476666667	0.0509166667
		9	0.0575	0.056	0.049	0.0541666667	

Table 11: Shi-Tomasi Timing

RESOLUTION	CONTRAST	QUALITY	DISTANCE	TIME 1	TIME 2	TIME 3	AVG	AVG @ CONTRAST
							0.060333333	0.059083333
480	0.01	0.01	5	0.061	0.063	0.057	3	3
			25	0.078	0.079	0.072	0.076333333	

							3	
							0.050666666	
		0.4	5	0.055	0.049	0.048	7	
			25	0.051	0.052	0.044	0.049	
							0.050333333	
	0.3	0.01	5	0.051	0.053	0.047	3	0.0555
							0.068666666	
			25	0.071	0.066	0.069	7	
		0.4	5	0.051	0.044	0.046	0.047	
			25	0.058	0.059	0.051	0.056	
							0.093333333	0.105666666
1560	0.01	0.01	5	0.091	0.094	0.095	3	7
							0.106333333	
			25	0.108	0.105	0.106	3	
							0.106333333	
		0.4	5	0.109	0.108	0.102	3	
							0.116666666	
			25	0.119	0.115	0.116	7	
							0.100333333	
	0.3	0.01	5	0.098	0.102	0.101	3	0.10965
						0.109		
			25	0.112	0.11	8	0.1106	
							0.110666666	
		0.4	5	0.116	0.109	0.107	7	
			25	0.118	0.119	0.114	0.117	

Harris and Shi-Tomasi are compared at the contrast level. With a 480p image given contrast threshold 0.01, Harris outperforms Shi-Tomasi. At the same resolution, but with a contrast threshold 0.3, there is a similar outcome, though it is a much closer race here. At 1560p, Harris appears to perform 20 times better than Shi-Tomasi on a speed basis. Though, again we see that despite a jump from 480p to 1560p, the timing of the Harris corner detector seems unchanged, despite the jumpy in the number of pixels to process. For lower resolutions, it is clear that Harris has faster performance than Shi-Tomasi, but the unchanged nature of Harris when going from 480p to 1560p warrants neglecting to conclude anything about Harris vs Shi-Tomasi at higher resolutions.

Keypoint Quantity: The testing scheme described above was run on every other image from the simulated data set, at 480p resolution. The following data compares the first, middle, and last images tested, for brevity's sake.

img20080	**Table 12: AVG # KEYPOINTS**		
INTV / OCTV	**SIFT**	**ORB**	**SURF**
1	106	186	58
2	116	312	112
3	125	373	138
4	122	408	156

5	135	427	173
6	135	439	184
7	136	444	195
8	147	446	205
9	155	446	209
10	176	446	216
11	196	445	223
12	232	443	229
13	258	442	232

Here, the data shows that ORB outperforms SIFT and SURF with regards to the quantity of keypoints produced at each interval parameter. Notice that ORB and SURF seem to be reaching a level off point for quantity of keypoints. This kind of trend is an example of how parameter extrema were chosen. Another interesting point is to note the rate of increase in keypoint quantity. With SIFT, the tendency is to slightly linearly increase the rate of keypoint quantity production as the number of intervals per octave increases. ORB shows a very different story, as the tendency appears to be to exponentially decrease the rate of keypoint quantity production as the number of intervals increases. Though the difference for SURF is dramatic at first, the overall tendency is to slightly linearly decrease the rate of keypoint quanityt production as the number of intervals per octave increases. Though ORB outperforms both SIFT and SURF, it is worth noting that SIFT and SURF produce a similar quantity of keypoints

img20100	**Table 13: AVG # KEYPOINTS**		
INTV / OCTV	**SIFT**	**ORB**	**SURF**
1	54	92	0
2	56	129	52
3	64	163	64
4	71	186	69
5	84	205	79
6	100	221	83
7	114	234	87
8	119	243	94
9	138	249	96
10	156	255	98
11	197	259	102
12	260	261	104
13	272	264	104

With this image, which is more overhead than the other two, shows ORB outperforming both SIFT and SURF, though the margin is much narrower this time from the previous time between ORB and SIFT. With regards to rate of increase of keypoint quantity, SIFT seems to be following a reverse trend than before, now linearly slightly linearly increasing the rate of keypoint quantity. ORB follows a similar trend to before, but this time less dramatically. SURF appears to be following almost exactly the same trend. It is worth noting here that though ORB outperforms both SIFT and SURF, SURF never comes close to SIFT, as it did in the previous image.

img20118	**Table 14: AVG # KEYPOINTS**		
INTV / OCTV	**SIFT**	**ORB**	**SURF**
1	58	104	32
2	52	177	51
3	57	232	62
4	72	274	76
5	78	306	87
6	86	329	92
7	92	342	98
8	90	352	104
9	100	359	108
10	116	362	114
11	139	363	123
12	185	364	127
13	202	365	129

Here, ORB again outperforms both SIFT and SURF. Though the angle of incidence is similar to the first image, SURF still lags behind SIFT. The rate of increase of keypoint quantity is very unsteady for SIFT with this image, seemingly randomly increasing or decreasing the rate of keypoint quantity production. ORB shows a similar trend to the previous two images, almost back to exponentially decreasing the rate of increase of keypoint quantity. SURF again shows slight linear decrease in the rate of increase of keypoint quantity production.

img20080	**Table 15: AVG # KEYPOINTS**	
CONTRAST	**Harris**	**Shi-Tomasi**
0.01	33	24
0.03	32	24
0.05	31	24
0.07	28	24
0.09	26	24
0.11	22	24
0.13	19	24
0.15	14	24
0.17	11	24
0.19	9	24

0.21	9	24
0.23	9	24
0.25	15	24
0.27	0	24
0.29	0	24
0.31	0	24

Though Harris and Shi-Tomasi are supposed to be compared at the contrast level (as the implementation from OpenCV both provide a contrast parameter), the steadiness of Shi-Tomasi make the results seem like they can't be compared at this level, as the contrast threshold parameter does not seem to change the average number of corners for Shi-Tomasi. One thing to note is that the Harris corner detector generally gradually decreases the quantity of corners as the contrast threshold increases.

img20100	**Table 16: AVG # KEYPOINTS**	
CONTRAST	**Harris**	**Shi-Tomasi**
0.01	19	18
0.03	19	18
0.05	20	18
0.07	22	18
0.09	23	18
0.11	22	18
0.13	18	18
0.15	18	18
0.17	17	18
0.19	13	18
0.21	10	18
0.23	10	18
0.25	9	18
0.27	0	18
0.29	0	18
0.31	0	18

A similar story can be said with this image as with the last image, though it is also interesting to not that the average number of corners produced by the Harris detector rises slightly as the contrast threshold begins to increase, but then falls back again with increased contrast threshold.

img20118	**Table 17: AVG # KEYPOINTS**	
CONTRAST	**Harris**	**Shi-Tomasi**
0.01	24	18
0.03	22	18
0.05	20	18
0.07	19	18
0.09	18	18
0.11	17	18
0.13	16	18
0.15	15	18
0.17	17	18
0.19	18	18
0.21	16	18
0.23	14	18
0.25	15	18
0.27	0	18
0.29	0	18
0.31	0	18

This dataset show a similar result to the first image, with Shi-Tomasi remaining constant across contrast thresholds while the number of corners produced by the Harris detector generally gradually decreases.

Keypoint Quality: Though sample images were produced at parameter extrema, these ended up producing 0 keypoints for the ORB algorithm. Despite this, the "best" images, with regards to the spread of keypoints and closeness of keypoints to the objects of interests are show below. All images are at a resolution of 480p.

Figure 1: SIFT – 13 intervals per octave, edge threshold 0, contrast threshold 0.1

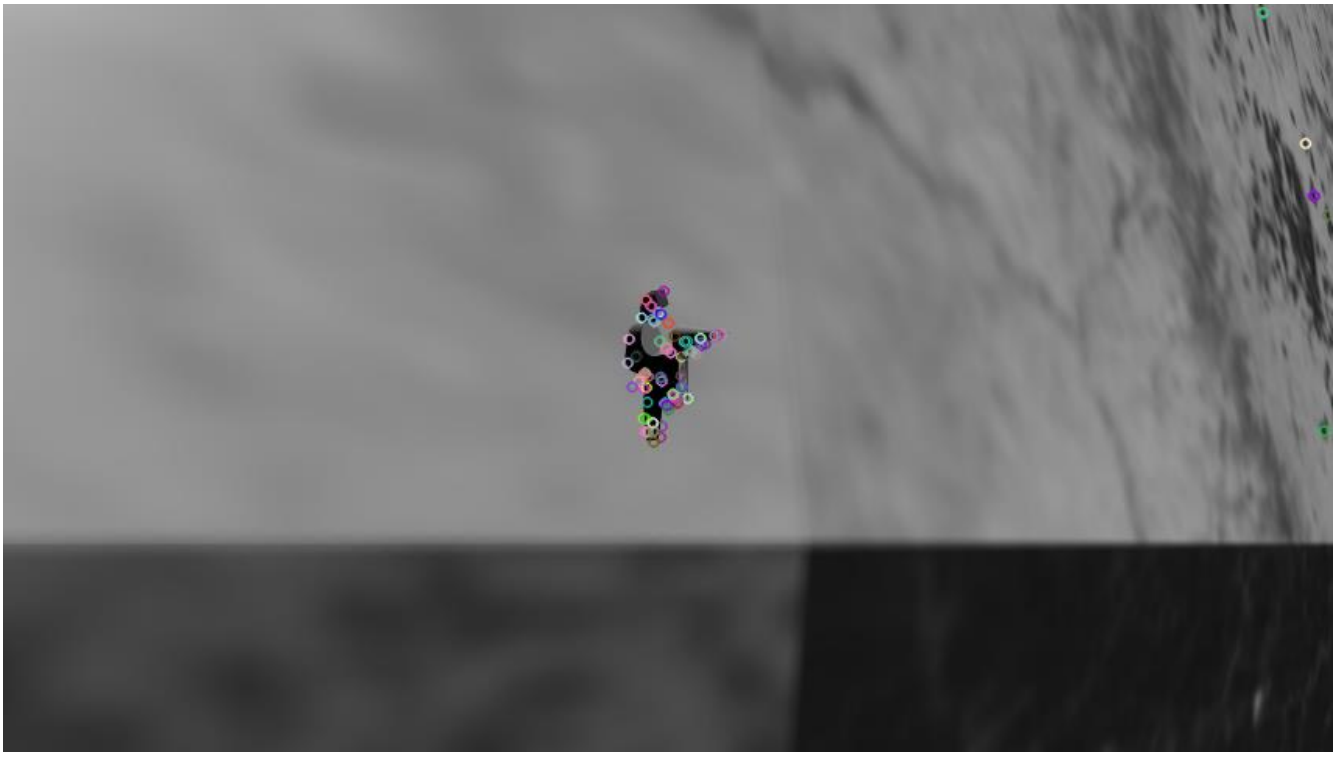


Figure 2: SIFT – 13 intervals per octave, edge threshold 24, contrast threshold 0.1

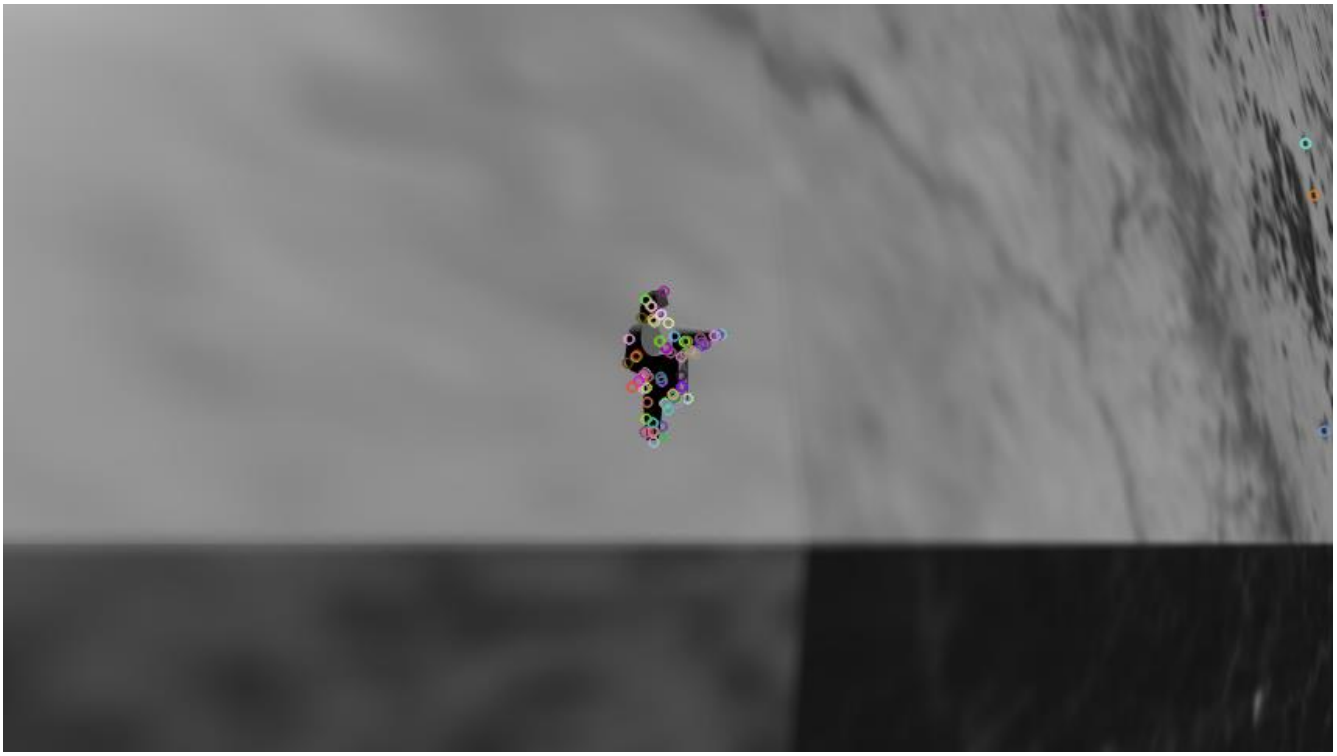


Figure 3: ORB – 7 intervals per octave, edge threshold 120, FAST threshold 21

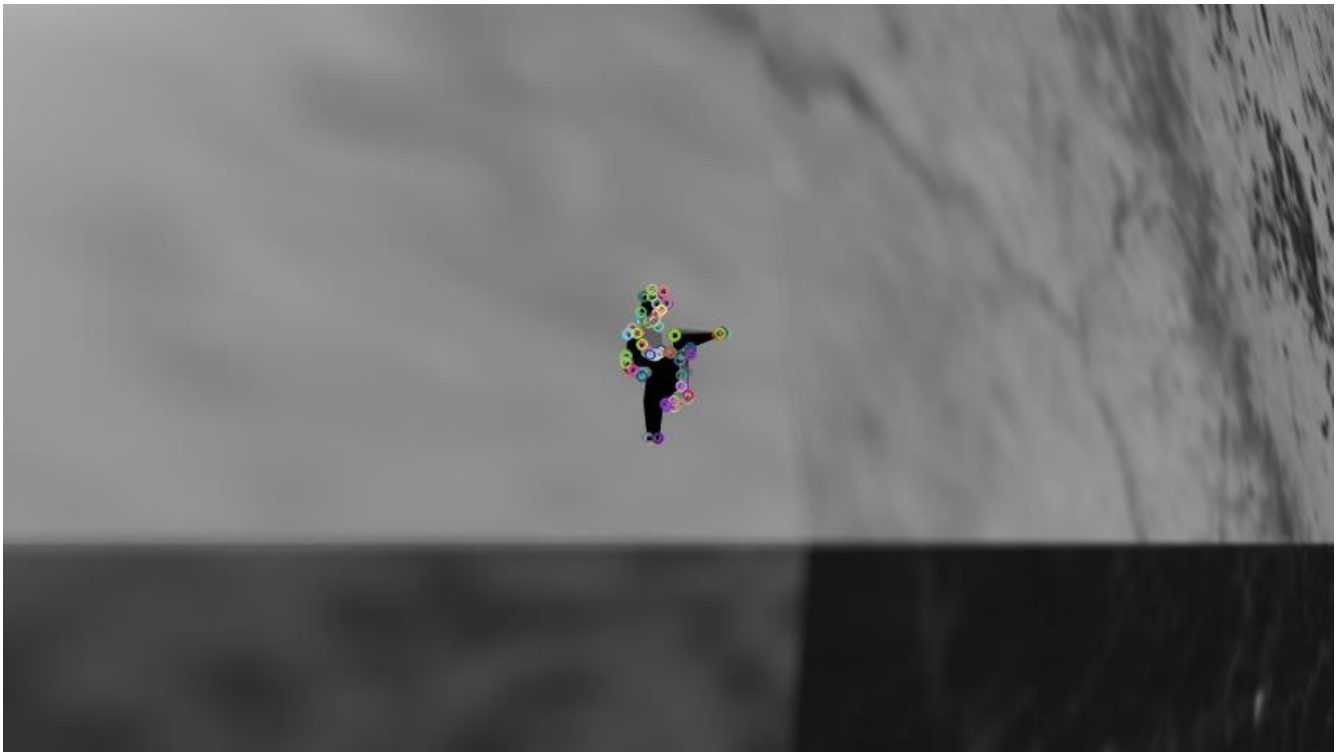


Figure 4: ORB – 13 intervals per octave, edge threshold 0, FAST threshold 41

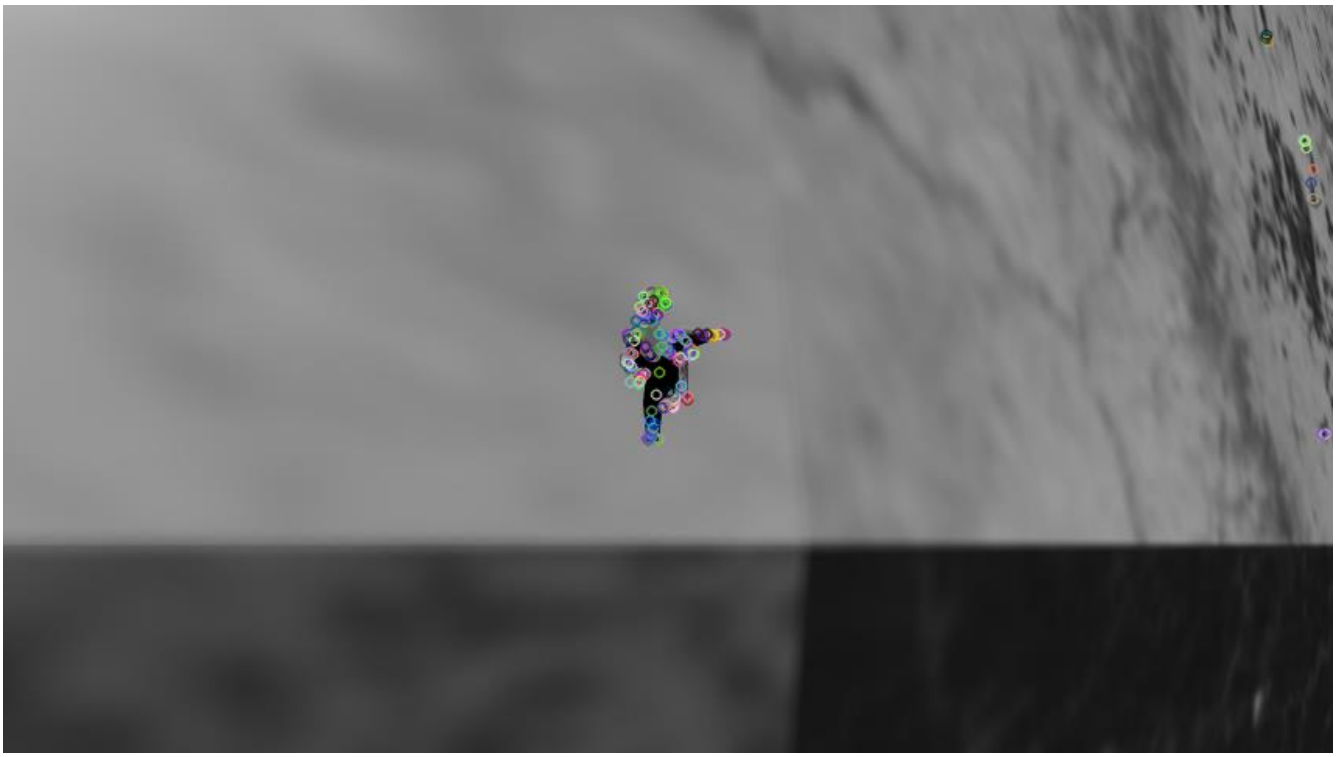


Figure 5: SURF – 1 interval per octave, Hessian threshold 505

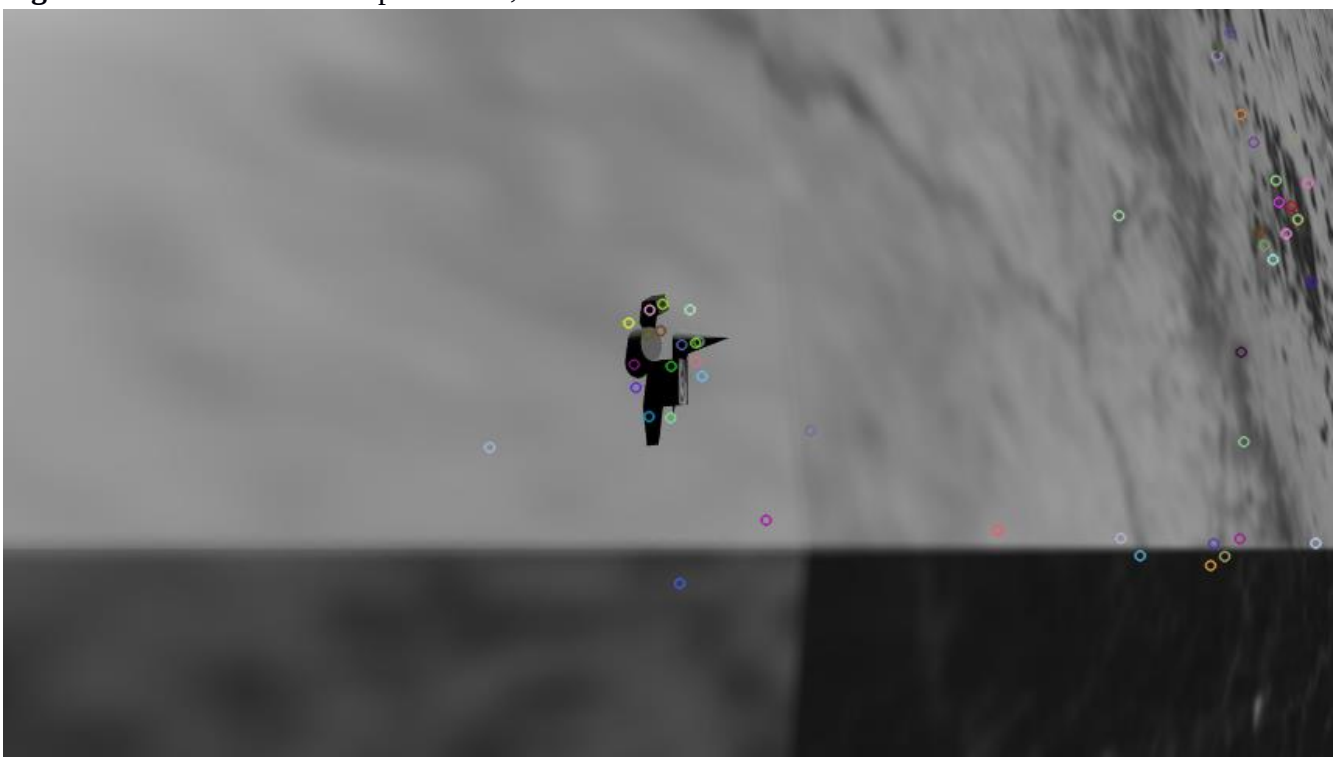
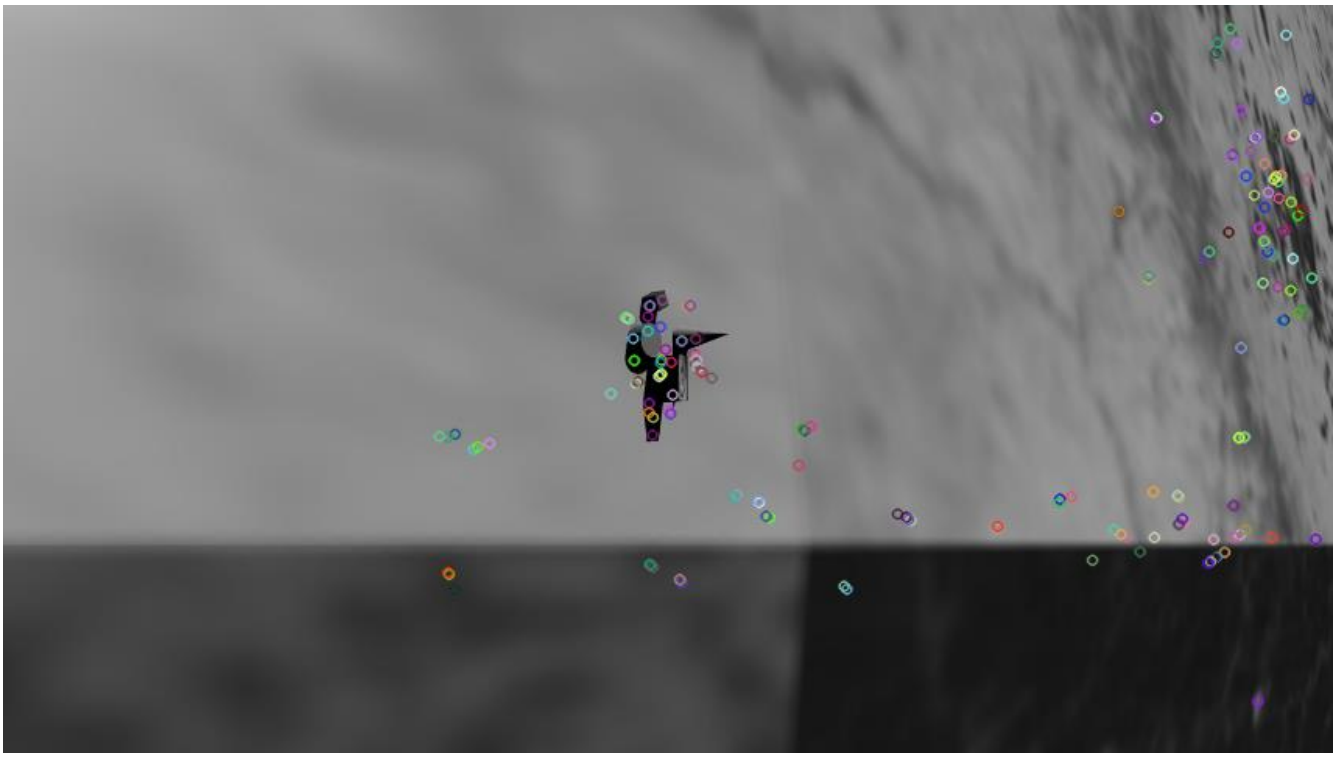


Figure 6: SURF – 13 intervals per octave, Hessian threshold 505



From these images, it can be seen that the ORB keypoint detector perform at about the same level as the SIFT keypoint detector with regards to having a low spread of keypoints and have a high number of keypoints being near the objects of interest. Though SURF produces more keypoints than the other two in these images, the keypoints are more spread out through the image. SIFT seems slightly more likely than ORB to assign keypoints not near the objects of interest, but this detail is negligible.

Figure 7: Harris – contrast threshold 0.01, Sobel apreature 1, corner threshold 0.64

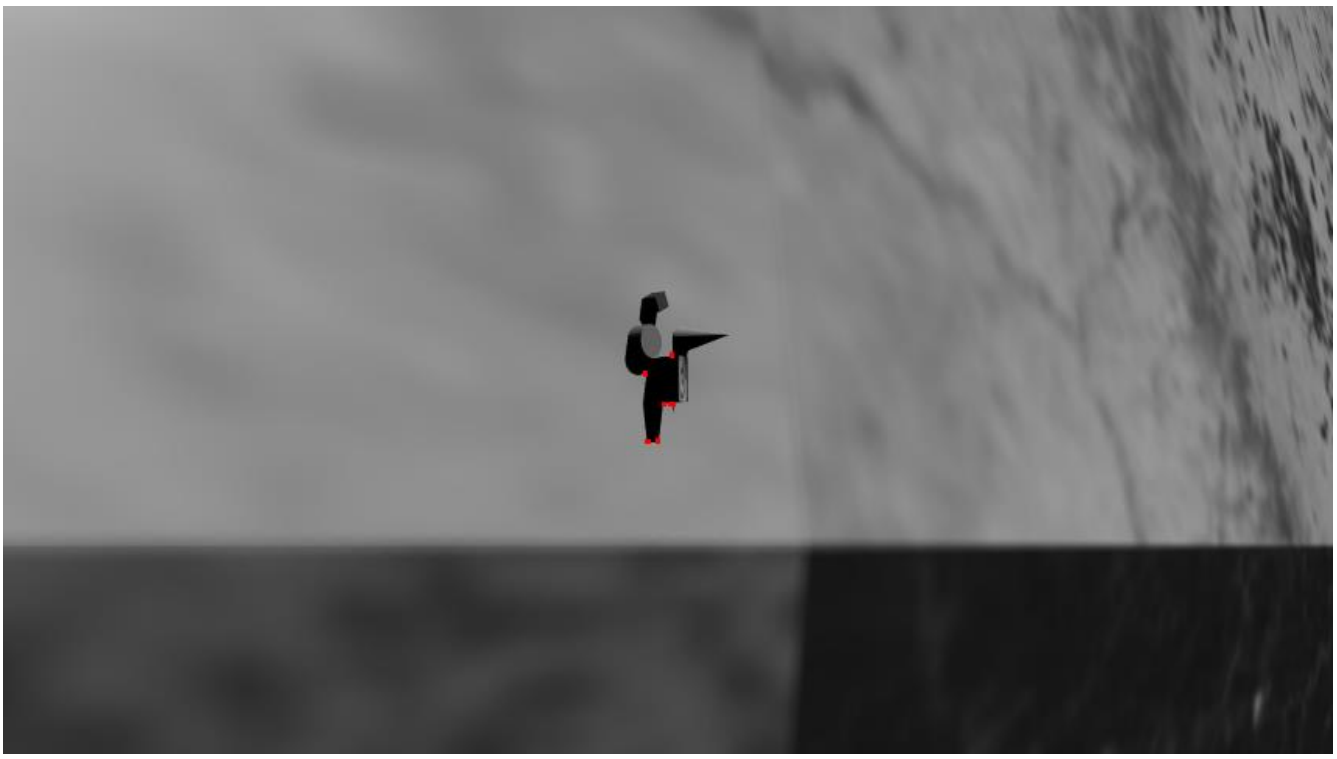


Figure 8: Harris – contrast threshold 0.01, Sobel apreature 9, corner threshold 0.64

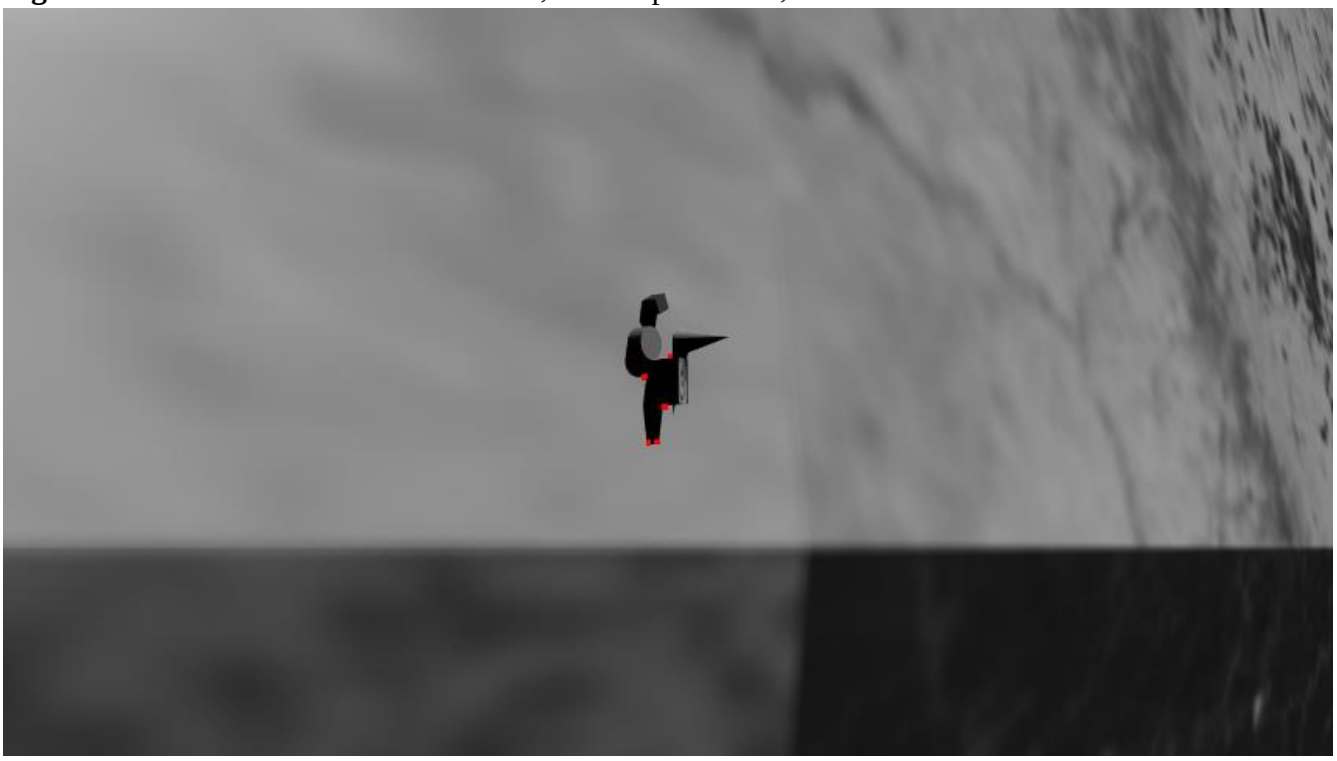


Figure 9: Shi-Tomasi – contrast threshold 0.01, corner quality 0.11, minimum corner separation 5

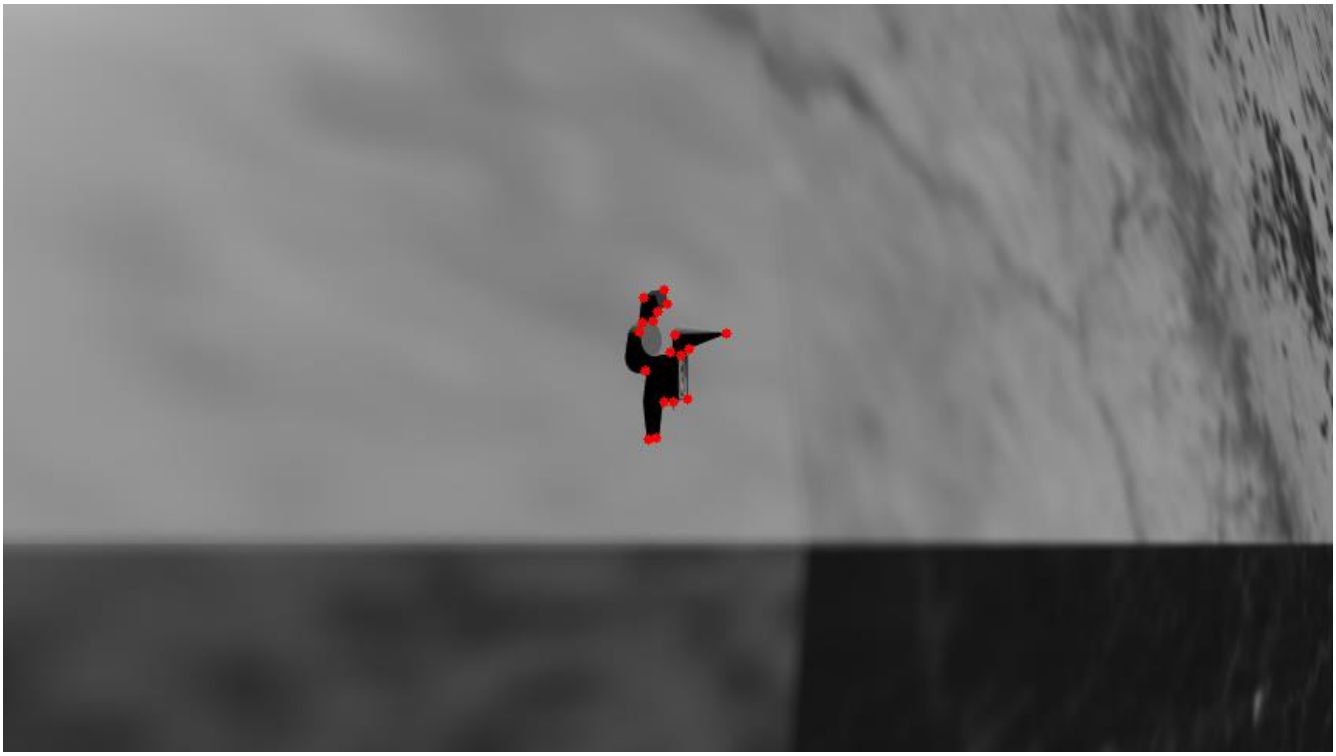
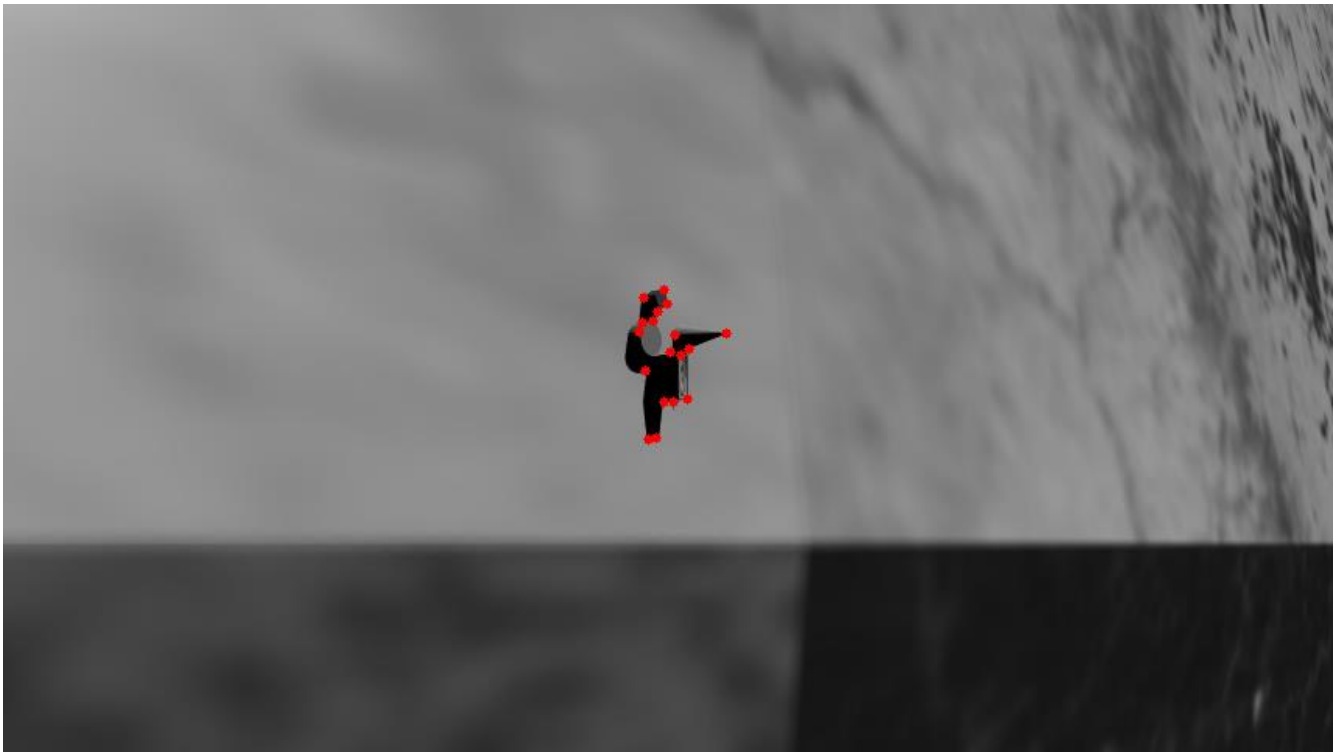


Figure 10: Shi-Tomasi – contrast threshold 0.32, corner quality 0.11, minimum corner separate 5



This data shows that both Shi-Tomasi seems to perform better than Harris with regard to only the number of corners near the objects of interest, as neither Harris nor Shi-Tomasi have a high spread of corners throughout the image. The Harris corner detector also seems to only capture corners where there is a stark contrast, while the Shi-Tomasi picks up corners that are along object edges.

Conclusion

Due to some unknown errors with the ORB timing tests, perhaps due to misunderstanding of the algorithm, no conclusions can be made to compare ORB's timing to SIFT's or SURF's, as there appears to be no difference in computation time when jumping from a 480p image to a 1560p image. With this, it can still be said that SURF outperforms SIFT in the speed category. ORB produces a larger number of keypoints than both SIFT and SURF, so the ORB algorithm comes out on top in the keypoint quantity category. With the quality of keypoints, the SURF algorithm had a large spread of keypoints across the image, whereas the ORB and SIFT algorithms had keypoints focused on the objects of interest. There was no noteworthy gain when using ORB or SIFT here, so these two algorithms tie for keypoint quality. With this, it can be concluded that in general, ORB is superior to both SIFT and SURF. This summary can be visualized in the table below:

Table 18	**SIFT**	**ORB**	**SURF**
Timing	0	0	1
Keypoint Quantity	0	1	0
Keypoint Quality	1	1	0
Sum	1	2	1

Also due to some unknown errors with the Shi-Tomasi average number of corners testing, again perhaps due to misunderstanding of the algorithm, no conclusions can be made to compare Harris' and Shi-Tomasi's quantity of keypoints. Though Harris seemed superior to Shi-Tomasi with the timing tests, the results were too close to say anything absolute, making the category of speed a tie for these two algorithms. In the above data, it is clear that Shi-Tomasi has a higher number of keypoints around the objects of interest, so it is superior to Harris in this category. With this, it can be concluded that in general, the Shi-Tomasi corner detector is superior to the Harris corner detector. This summary can be visualized in the table below:

Table 19	**Harris**	**Shi-Tomasi**
Timing	1	1
Keypoint Quantity	0	0
Keypoint Quality	0	1
Sum	1	2

Future Directions

As discussed above, there were problems with the ORB algorithm with regards to testing timing of keypoint detection. Though the jump from a 480p image to a 1560p image involves increasing the pixel count by ~10.5 times, the timing for ORB remained relatively unchanged. This could be attributed to a misunderstanding of the algorithm, so further research into this could solve this issue.

The original plan for exhaustive testing to determine the average number of keypoints produced by each algorithm involved running each algorithm on varied image resolution, from 480p up to 1560p at a step size of 120. An initial run of this version of testing halted for some unknown reason, and though the reason for the halting was never solved, it did bring to attention one issue: the amount of time required to run each algorithm on higher and higher resolution images. The next round of exhaustive testing was performed on only images using 480p resolution, and the UNIX time function was used to estimate how long the exhaustive testing would run if performed on 1560p resolution images. This test was of 16 images at 480p, and resulted in a system time of 21m25.608s and a user time of 140m13.628s. The full suite of exhaustive testing, which involved going through every parameter permutation for SIFT, ORB, and SURF, had an estimated system time of ~80s per image at 480p. It was estimated that this process on 600p images would take ~125s of system time per image, and on 720p images would take ~180s of system time per image. When extrapolating to 1560p, the estimated system time was estimated to be ~847s per image. For 40 images, this one resolution was estimated to take ~9.4 hours of system time, or ~66 hours of user time. At 600p, 40 images are about ~1.3 hours of user time, and at 720p, 40 images are about ~2 hours of user time. To run the exhaustive testing on all resolution is roughly estimated roughly to have taken ~2 weeks of user time, which was infeasible given the amount of time left for testing. Thus, it was decided that the only resolution to be used for the average keypoint calculation would be 480p.

There were also problems with the Shi-Tomasi algorithm with regards to calculating the average number of keypoints. Though the contrast threshold was a parameter for the Shi-Tomasi algorithm provided by OpenCV, it appeared to have no output on the average after considering all permutations of the rest of Shi-Tomasi's parameters. This made Shi-Tomasi not be able to compare with Harris, as the common ground was the contrast threshold. This could be attributed to a misunderstanding of the algorithm, or to an error in code, as the OpenCV project is opensource. Further research into this could solve the issue.

Further work could also include considering the storage space of each keypoint as a comparison criteria, along with the timing of keypoint descriptors and keypoint matching. A branch of this research could also include mixing and matching various keypoint detection, description, and matching algorithms to determine the best combination of algorithms that would produce an optimal output.

Acronyms

1.	AFRL	Air Force Research Laboratory
2.	BRIEF	Binary Robust Independent Elementary Features
3.	CPU	Central Processing Unit
4.	FAST	Features from Accelerated Segment Test
5.	LEO	Low-Earth Orbit
6.	MOCI	Mapping and Ocean Color Imager
7.	MU-SURF	Modified Upright SURF
8.	ORB	Oriented FAST and Rotated BRIEF
9.	SFM	Structure From Motion
10.	SIFT	Scale-Invariant Feature Transform
11.	SSRL	Small Satellite Research Laboratory
12.	SURF	Speeded-Up Robust Transform
13.	UGA	University of Georgia
14.	UNP	Undergraduate Student Instrument Project

References

[1] Adams, Caleb. “The Feasibility of Structure from Motion over Planetary Bodies.” Ed. David Cotten. SSRL Internal Documentation. 9 Mar. 2017.

[2] Bay, Herbert, et al. "Speeded-up robust features (SURF)." *Computer vision and image understanding* 110.3 (2008): 346-359. 9 Mar. 2017.

[3] Ebrahimi, Mosalam, and Walterio W. Mayol-Cuevas. "SUSurE: Speeded up surround extrema feature detector and descriptor for realtime applications." *Computer Vision and Pattern Recognition Workshops, 2009. CVPR Workshops 2009. IEEE Computer Society Conference on*. IEEE, 2009. 13 Mar. 2017.

[4] Ilango, Nirav. “MOCI Mission Overview.” Ed. David Cotten. SSRL Internal Documentation. 9 Mar. 2017.

[5] Jebara, Tony, Ali Azarbayejani, and Alex Pentland. "3D structure from 2D motion." *IEEE Signal processing magazine* 16.3 (1999): 66-84. 9 Mar. 2017.

[6] Lowe, David G. "Distinctive image features from scale-invariant keypoints." *International journal of computer vision* 60.2 (2004): 91-110. 9 Mar. 2017.

[7] Rublee, Ethan, et al. "ORB: An efficient alternative to SIFT or SURF." *Computer Vision (ICCV), 2011 IEEE International Conference on*. IEEE, 2011. 12 Mar. 2017.

[8] Schonberger, Johannes L., and Jan-Michael Frahm. "Structure-from-motion revisited." *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2016. 9 Mar. 2017.

[9] Jacob Toft Pederson. “SURF: Feature Detection and Description.”, 2011.

[10] Luo Juan, Oubong Gwun. “A Comparison of SIFT, PCA-SIFT and SURF”, *International journal of Image Processing(IJIP),* Volume(3) Issue(4).

[11] Alexandre Alahi, Raphael Ortiz, Pierre Vandergheynst. “FREAK : Fast Retina Keypoint” CVPR '12 Proceedings of the 2012 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), June 2012.

[12] Dr. Mubarak Shah. “Interest Point Detection”, University of Central Florida, 2012. http://crcv.ucf.edu/courses/CAP5415/Fall2012/Lecture-4-Harris.pdf

[13] Chris Harris & Mike Stephens, “A Combined Corner and Edge Detection.” UK, 1998.

[14] Jainbo Shi, Carlo Tomasi. “Good Features to Track”, IEEE Conference on Computer Vision and Pattern Recognition(CVPR94), Seattle, June 1994.