International Journal of Research in Engineering, Science and Management
Volume-3, Issue-1, January-2020
www.ijresm.com | ISSN (Online): 2581-5792

Protection Scheme for Star Tracker Images

S. Lingesh Siva

Student, Dept. of Electronics and Communication Engg., Marian Engineering College, Trivandrum, India

***Abstract*: Radiation effects in star tracker can cause malfunction that may lead in loss of communication with the satellite. This paper presents a scheme to protect star tracker images stored in memory against single event upsets (SEUs). Fault injection and resource utilization reports show that, with our protection technique,99.98% off the SEUs can be corrected using 37% less memory space than Single Error Correction (SEC) Hamming Codes. The remaining 0.02% has a negligible effect in the star identification algorithm.**

***Keywords*: Error correction, Image processing, Memory, Soft error, Star tracker.**

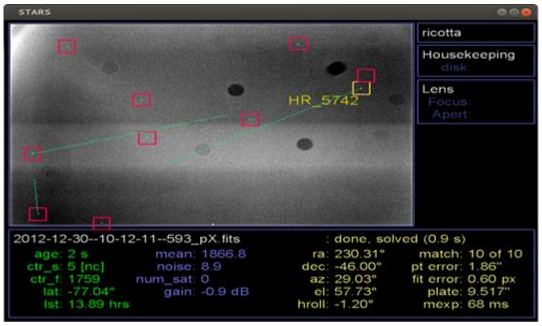
1. Introduction

Most of the satellites are usually equipped with attitude determination instruments such as star trackers. These high accuracy systems basically consist of camera that capture sky images, and an electronic processing device that capture sky images, and an electronic processing device that estimates the orientation of the spacecraft using the captured images. Each sky image is processed to extract the coordinates of the star centroids on the focal plane. The position of the star centroids is compared against a stored star catalogue, and the attitude quaternion values are calculated to rectify the current satellite orientation if necessary. These steps are automatically performed by the electronics processing system, which can be made of a microprocessor, or a combination of microprocessor and Field Programmable Gate Array (FPGA).

2. Star Tracker

A star tracker is an optical device that measures the positions of stars using photocells or a camera. As the positions of many stars have been measured by astronomers to a high degree of accuracy, a star tracker on a satellite or spacecraft may be used to determine the orientation (or attitude) of the spacecraft with respect to the stars. In order to do this, the star tracker must obtain an image of the stars, measure their apparent position in the reference frame of the spacecraft, and identify the stars so their position can be compared with their known absolute position from a star catalogue.

A star tracker may include a processor to identify stars by comparing the pattern of observed stars with the known pattern of stars in the sky.



```
2012-12-30--10-12-11--593_pX.fits                : done, solved (0.9 s)
   age: 2 s          mean: 1866.8          ra: 230.31°       match: 10 of 10
 ctr_s: 5 [nc]      noise: 8.9            dec: -46.00°     pt error: 1.86"
 ctr_f: 1759      num_sat: 0               az: 29.03°     fit error: 0.60 px
   lat: -77.04"       gain: -0.9 dB         el: 57.73°        plate: 9.517"
   lst: 13.89 hrs                        hroll: -1.20°         mexp: 68 ms
```

Fig. 1. The STARS real-time star tracking software operates on an image from EBEX 2012, a high-altitude balloon-borne cosmology experiment launched from Antarctica on 2012-12-sa

3. Current technology

Many models are currently available. Star trackers, which require high sensitivity, may become confused by sunlight reflected from the spacecraft, or by exhaust gas plumes from the spacecraft thrusters (either sunlight reflection or contamination of the star tracker window). Star trackers are also susceptible to a variety of errors (low spatial frequency, high spatial frequency) in addition to a variety of optical sources of error (spherical aberration, chromatic aberration, etc.). There are also many potential sources of confusion for the star identification algorithm. There are roughly 57 bright navigational stars in common use. However, for more complex missions, entire star field databases are used to determine spacecraft orientation. A typical star catalog for high-fidelity attitude determination is originated from a standard base catalog (for example from the United States Naval Observatory) and then filtered to remove problematic stars.

A. *Star Camera*

Most spacecraft steer by the stars. Or, to make sure they are still on course and pointed in the right direction, they may periodically check the position of the stars. So, what is a star camera often referred to as a star tracker. A star camera or star tracker is a "celestial reference" device that recognizes star patterns, such as constellations. Star patterns, and even single stars, are very helpful for navigation. In ancient times, sailors navigated by the North Star it was their reference point. By looking at it they could tell if they were on course.

This is similar to checking the position of our sun, which rises in the east and sets in the west, and being able to tell in which direction we are heading. The sun gives us point of reference a starting point that we refer to and compare our position to as we travel.

To recognize star patterns, Compass uses an active pixel sensor (APS) in a wide-field-of-view (WFOV) miniaturized star camera. APS is a new type of compact imaging device with an array of photo sensors. APS uses a fraction of the power used by standard charged coupled devices, enabling a major reduction in the power, size, weight, and cost of imaging and spectroscopy instruments. This makes APS very attractive for use on small, low power spacecraft and instruments. The Compass WFOV star camera will observe a wider slice of sky, taking pictures of the star patterns in its view. And like a star tracker, the star camera will locate the positions of stars and report them to the spacecraft. The captured images will then be compared to a celestial map that resides in the spacecraft computer memory. The Compass star camera's sensor will track both bright and dim objects in its field of view. It will also prevent extremely bright objects, such as the moon or Earth, from spilling over into the pixels of captured objects.

Compass star camera sends data to the gyroscopes, which can hold stable for just a short time, every few seconds. This helps to keep the gyroscopes accurate. Together, the star camera and gyroscopes keep the spacecraft stable and oriented in the right direction in space.

4. Related Works

The scheme is divided into two parts.

A. Star tracker data flow

Microprocessor can be used together with a FPGA to implement star tracker system. The FPGA handles the data acquisition and the image processing part, and stores the pre-processed image in the memory. Then, the microprocessor reads the stored image and performs the centroid calculation and the star matching algorithm to determine the satellite attitude.

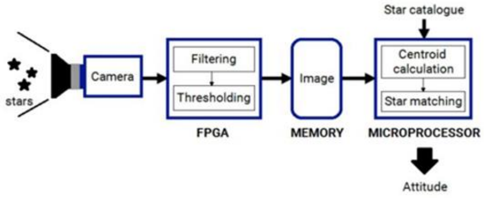


Fig. 2. Data flow diagram of star tracker based on FPGA and microprocessor

The FPGA exploits the inherent parallelism in image to accelerate the image pre-processing operations, while the microprocessor enables high performance when executing complex matching algorithms. The star tracker data flow from the image sensor to the attitude determination step. The camera captures the raw image that is pre-processed by the FPGA. The pre-processing step usually consists of the filtering and thresholding modules as shown in fig.4.1. The filtering module minimizes the undesirable noise captured by the image sensor, and the thresholding operation sets to zero (i.e., black) those pixels in the image that are below a chosen value, leaving the star pixels unaltered. The pre-processed image is then stored in an external or internal memory that is shared with the microprocessor. Once the stored image is read by the microprocessor, it extracts the star centroids. The satellites attitude is determined by comparing this information against an onboard star catalogue database. In order to transfer the pre-processed image frame from the FPGA to the microprocessor, memory elements such as SRAM or Xilinx block RAMs (BRAMs) can be used. If the memory space available is not enough to store the whole image, it can be compressed using different algorithms to reduce the amount of data stored in the on-board memory. However, this strategy requires a decompression process during the read operation, so the algorithm running in the microprocessor become even more complex. Image compression algorithm also requires all or part of the image to be temporarily stored before applying them. Consequently, the pixels are vulnerable to space radiation while they are stored in memory, so an error correction technique is necessary.

B. Single Error correction (SEC) approaches for memories.

Hamming-based SEC codes are popularly used to protect memories and circuits from radiation-induced single even upsets (SEUs). In SEC Hamming codes (n-k) parity bits are added to the k-bit data a word, forming a new word of n bits.

For example, in the case of an 8-bit pixel, 4 extra parity bits are required to perform single error detection and correction. A SEC Hamming code uses a generator matrix (G) and parity-check matrix (H) that are defined as

$$G= (Ik \mid AT)$$

$$H = (A \mid I\ n\text{-}k)$$

Where Ik is the k×k identity matrix and A is an $(n - k) \times k$ matrix constructed by listing k columns of length $(n - k)$ that are pair-wise independent. In our case, the 8-bit pixel is encoded by multiplying it by G before storing it in memory along with its 4 extra parity bits. After the read operation, the pixel is decoded by multiplying it by H, obtaining a vector called syndrome. Then, if the syndrome is the null vector, it means that there is no error in the read pixel. Otherwise, the syndrome vector can be used to determine and correct the erroneous bit.SEC Hamming codes have minimum Hamming distance of three, so double errors can be mis corrected. To avoid that, Single Error Correction Double Error Detection (SEC-DED) codes can be chosen.

Previous works have focused their efforts on reducing the encoding/decoding FPGA resources to minimize the probability of SEUs affecting the design. In Xilinx BRAMs, the content of the memories is particularly sensitive to radiation due to their high density in the newer 7-series technologies. For this reason, the stored data can also be corrupted by SEUs.

Consequently, the minimization of the information stored in memory will lead to fewer data corruption.

5. Proposed Protection Scheme

The number of stars in the camera field of view (FOV) and their brightness are important for the accuracy of the centroid calculation algorithm. The number of stars per image can be calculated using equation

$$Ns = 6.57.\ e1.08Mv.\ ((1-\cos((FOV)/2)/2)$$

The number of stars Ns is usually between 4 and 6, depending on the FOV size (in degrees) and the visual magnitude (MV) of the star tracker. Moreover, each star is spread over a small region of 9 X 9 pixels since the optics in the star tracker are slightly defocused on purpose in order to achieve sub pixel centroiding accuracy. Therefore, a typical 640 X 480-pixel sky image will normally contain less than 500-star pixels. In other words, the percentage of pixels that belongs to stars is less than 0.2% for this image size.

The proposed protection scheme exploits this small percentage by creating a back-up copy of the star pixels for future corrections. The amount of memory reserved for the back-up copy can be calculated as follows:

$$NBRAM\ s = (Ns.Np.Nb)/Sb$$

Where Ns is previously calculated number of stars in the image
Np is the number of pixels per star,
Nb is the number of bits per pixel, and Sb is the size in bits of one BRAM or memory element.

A. *Encoding operation*

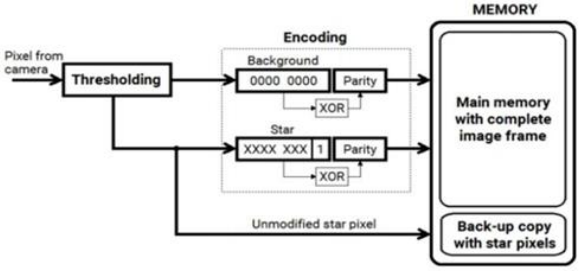


Fig. 3. Proposed protection scheme (encoding operation)

The encoding operation illustrated in fig 5.2 is performed for each pixel. First, the pixel captured by the camera is classified as a star or a background pixel by the thresholding module already present in a typical star tracker system. If the pixel is below the threshold, it is considered background and set to zero in the encoding stage, Then, the encoded background pixel is stored in the main memory along with its parity, which is always zero. Conversely, if the pixel is above the threshold, it is considered star and, in our technique, the pixel is duplicated. One copy of the star pixel is stored unmodified in the back-up memory and, at the same time, the other copy is encoded and stored in the main memory. it should mention that this replacement of the LSB by '1' in the star pixel case adds a small noise of +1 to the star intensity in some cases, however, its impact in the centroid calculation is within the inherent tolerance of the star identification algorithm. With the proposed encoding approach, a minimum Hamming distance of three between a star and a background pixel is created, so the capability to differentiate between a corrupted star pixel and a corrupted background pixel during the read operation is enabled. A Hamming distance of three means that the number of bit changes needed to convert any star pixel to a background pixel (or vice versa) is at least three. As mentioned before, any background pixel is encoded as all zeros, so its parity is also zero. Therefore, the Hamming distance in this case is equal to the number of ones in the star pixel. Since a star pixel is, by definition, a pixel that is above a certain threshold value, the worst-case scenario occurs when the star pixel only contains a '1' in the seven MSBs.

In order to achieve a Hamming distance of three, two additional ones are required. The second '1' is obtained by replacement of the star pixel LSB by '1', and the third '1' is created in the parity bit by the XOR operation of the seven MSBs (there is odd number of ones, so XOR is 1). In the remaining cases in which the number of ones in the seven MSBs is equal or greater than two, the Hamming distance of three is automatically achieved by the replacement of the LSB, or just by the total number of ones in the seven MSBs. To illustrate the encoding process, let us consider the codification of pixel 1000 0000 when the threshold value is 0100 0000. After the threshold operation, the value of the pixel is sent to encoding module. First, the parity bit of the seven MSBs is calculated (1 in this case), afterwards the LSB is replaced by '1' due to the pixel value being above the threshold.

B. Decoding operation

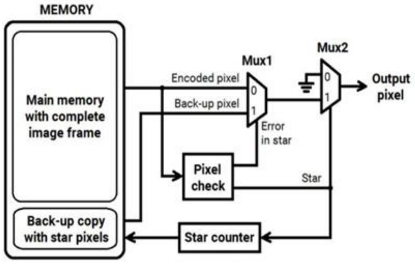


Fig. 4. Proposed protection scheme (Decoding Operation)

In order to correct radiation-induced single event upsets in the stored pixels, some checks have to be carried out during the read operation (see in fig. 4).

First, the “pixel check” module depicted in fig. 3 determines whether the read pixel is background or star. If pixel is background, the “star” signal connected to the second multiplexer is not triggered, and then the output pixel is set to zero, thus removing every possible SEU in the background pixel. Otherwise it is a star, and then the parity of its seven MSBs is checked. If the parity checking is correct, the star pixel is outputted normally. But in case of parity mismatch, the “Error in star” signal is triggered and the copy of the current star pixel is read from the back-up memory, in order to know the current back-up memory address, a star counter is required. This counter uses the “star” signal to increment the current back-up memory address.

For example, an SEU affecting a pixel stored in the back-up memory would not produce an erroneous output. This is because the back-up memory is only read when the pixel stored in the main memory is corrupted.

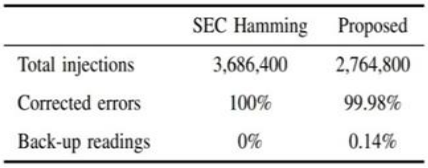
6. Implementation

This scheme implemented in the SRAM-based FPGA part of a Xilinx Zynq-7000 Zed Board. The error correction capability and memory and FPGA resource usage of both methods have been evaluated and are discussed in the following subsections.

A. Error Correction Capability

First, the proper operation of the proposed scheme and the SEC Hamming code is validated with different star tracker images. For the experiments, 8-bit grayscale 640 X 480 images have been generated using a MATLAB Program created By E. Palombo from the European Space Research and Technology Centre (ESTEC) in the Netherlands. Using these sky images, the “golden output” (i.e., the output in absence of error) is obtained for each protection scheme. Then, both schemes are tested in the presence of SEUs in the stored pixels. In order to perform error injection in the data, the input port INJECTSBITTER, which is inherently available in the 36Kb BRAM RAMB36E1 macro of the Xilinx 7 series FPGAs, has been used. By triggering this signal, single-bit errors can be injected. SEUs have been sequentially injected bit by bit and pixel by pixel for each test image. As can be observed in our fault injection layout illustrated in table 1, each pixel is encoded before storing it in memory by using our encoding approach or the SEC Hamming scheme depending on the technique under test. Then, and after every SEU injected, each output pixel is read, decoded, and compared to the golden pixel obtained in the no-error scenario. Finally, an error report is generated with this information.

Table 1
Error correction capability report



	SEC Hamming	Proposed
Total injections	3,686,400	2,764,800
Corrected errors	100%	99.98%
Back-up readings	0%	0.14%

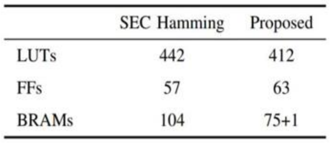
After the exhaustive fault injection campaign, the error correction capability of both SEC Hamming proposed scheme has been summarized, using the generated error reports. Results show that the proposed technique does not correct 0.02% of the injected errors. This is because, the decoding operation was configured to not correct errors in the LSB since their impact in the star intensity value and the centroid calculation in negligible. Therefore, the proposed technique can detect and correct all SEUs in background pixels and in the seven MSBs of star pixels, so the error coverage achieved for these bits is similar to SEC Hamming. In addition to the percentage of corrected errors, the back-up readings have also been taken into account, in this case, the SEC Hamming approach does not have a back-up, so the percentage is zero. However, in our proposed approach, a back-up reading has to be performed when one of the seven MSBs of a star pixel is detected as corrupted. The reduced amount of star pixels in the image decrease the number back-up readings, proving the effectiveness of our proposed approach.

Finally, it should be mentioned that the total number of injections performed for our scheme is lower than the SEC Hamming approach due to lower number of parity bits added to the unprotected pixel.

B. Memory usage and resource utilization

As stated previously, 4 parity bits are required to perform single error detection and correction in an 8-bit pixel using a SEC Hamming code. However, only one parity bit is necessary with the proposed scheme. Therefore, fewer memory addresses and thus less BRAMs are required to store the same 8-bit grayscale 640 X 480 image using the proposed scheme. This also implies that the counters used to increment the read/write addresses will require less FPGA resources. In order to prove that, a utilization report in terms of number of look-up tables (LUTs), flip-flops (FFs), and BRAMs has been obtained for each protection scheme. Both implementation reports are summarized in Table.

Table 2
FPGA resource and memory utilization report



	SEC Hamming	Proposed
LUTs	442	412
FFs	57	63
BRAMs	104	75+1

As observed from the table, the proposed scheme requires more FFs, but less LUTs and BRAMs than the SEC Hamming approach. Overall, the number of FFs and LUTs is similar in both designs but, in terms of BRAMs, the SEC Hamming approach needs up to 37% more memory space than the proposed scheme due to the higher number of redundant parity bits. It is worth mentioning that the board used in the experiment contains 53,200 LUTs, 106,400 FFs, but only 140 BRAMs. Therefore, the number of BRAMs is more restrictive than the number of LUTs or FFs

7. Benefits

The proposed protection scheme presents error correction capabilities that are consistent with the star tracker system. The proposed scheme can be used as an alternative to SEC Hamming codes, obtaining considerable savings in terms of memory usage. (37%) This scheme can also be used to protect other type of memories. This scheme can correct 99.98% of the SEU injected.

8. Conclusion

This scheme has been designed to exploit particular features of sky images to create a small back-up memory that contains a copy of the star pixels. If corrupted pixels are detected during the read operation, the type of pixel (background or star pixel) can be identified and then recovered.

References

[1] “Star Trackers". Goodrich, 2008.
[2] "Attitude and Orbit Control Systems". Jena-optronik.de.
[3] Hobbs, Marvin (2010). Basics of Missile Guidance and Space Techniques. Wildside Press. pp. 1–104.
[4] "Ball Aerospace star trackers". Ballaerospace.com
[5] “Attitude and orbit control system”. Jena-optronik.de.