A METHOD FOR THREE-AXIS ATTITUDE DETERMINATION BY IMAGE-PROCESSED STAR CONSTELLATION MATCHING

Hartmut Renken, H. J. Rath

University of Bremen, Center of Applied Space Technology and Microgravity (ZARM)
Am Fallturm, 28359 Bremen, Germany, E-mail: renken@zarm.uni-bremen.de

ABSTRACT

An unmanned system for automatically 3-axis attitude determination by star constellation matching is a very useful tool for applications primary in the field of satellite technology, where stars are visible all the time. A special designed CCD camera system, which is sensitive enough to detect stars, points to the direction (attitude) to be determinated and captures stars within the camera field of view (fov). By comparing this imaged star pattern with celestial coordinates out of a database-like star catalogue, a star pattern can be matched and the attitude (3-axis) can be calculated by using simple vector algebra. In partnership with other attitude determination systems, the accuracy and reliability for an attitude control system is increasing. Including camera hardware and image processing software, we realized a transputer based (*Inmos T8XX*, *Occam, Parallel C*) onboard computer system for our own satellite missions. For terrestrial and ground based pointing applications or post-mission attitude determination a powerful PC-based system (*Pentium*, *Pascal, C*) is available with additional visualization features of certain steps during the image processing via an userfriendly desktop.

Keywords: attitude determination, image processing, matching, pointing, star tracker

1. INTRODUCTION

A possibility to identify unknown star pattern by a computer-based system is described in this paper. One application is the 3-axis attitude determination of satellites (body z-axis and *third angle* of spacecraft) if the imaging system (CCD camera system) which detects the stars is orientated to a useful direction, preferred parallel to the spacecrafts attitude axis (fig. 1 and 2). The basic idea of matching is the geometrical correlation of a camera given star pattern with informations of celestial star positions via a star catalogue. The star catalogue contains all the stars, which are available for the CCD camera system limited by sensitivity and exposure time of the sensor. So there are two inputs for the matching system: the first input is a snapshot of a real celestial area and the second input is a star catalogue, containing the positions of stars (2-dimensional equatorial system (right ascension, declination) or preferably coordination transformed to a quasi 3-dimensional x-, y-, z- unit vector system) at the celestial sphere.

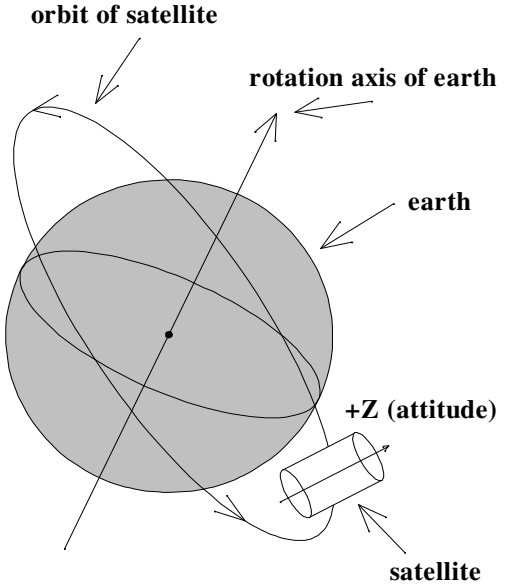

Fig. 1: Orbit of satellite around the earth with attitude (+z) to be determined

Because of the possibility to watch only the surface of the earth, the bright sun or a starless celestial area it is recommended to use more imaging systems with different orientations at the sky. So one useful image should be available in minimum. After checking the quality of the input image and deleting pixel disturbances and non-stellar objects if necessary, all celestial objects are segmentated. The segmentated objects are stored via data arrays with their area strong point (*x_obj*, *y_obj*) and brightness (*gray_value_obj*). Next is to assemble a pattern of stars with the brightest objects and try to find this imaged constellation by artificial reconstruction of this pattern, using only geometrical informations available via the star catalogue. If the matching was successful the attitude is calculated. If the optical axis of the CCD camera system is adjusted parallel to the attitude axis of the spacecraft, the corresponding celestial coordinates of the center of the image are calculated by simple vector algebra and linear combinations of the matched objects within the image and their corresponding celestial coordinates (2-axis attitude). At least the turn of the attitude z-axis itself is determinated by calculating the angle between the *third angle vector* (x-axis of image or body x-axis of spacecraft) and the celestial equator (3-axis attitude). See fig. 2 for more illustration.

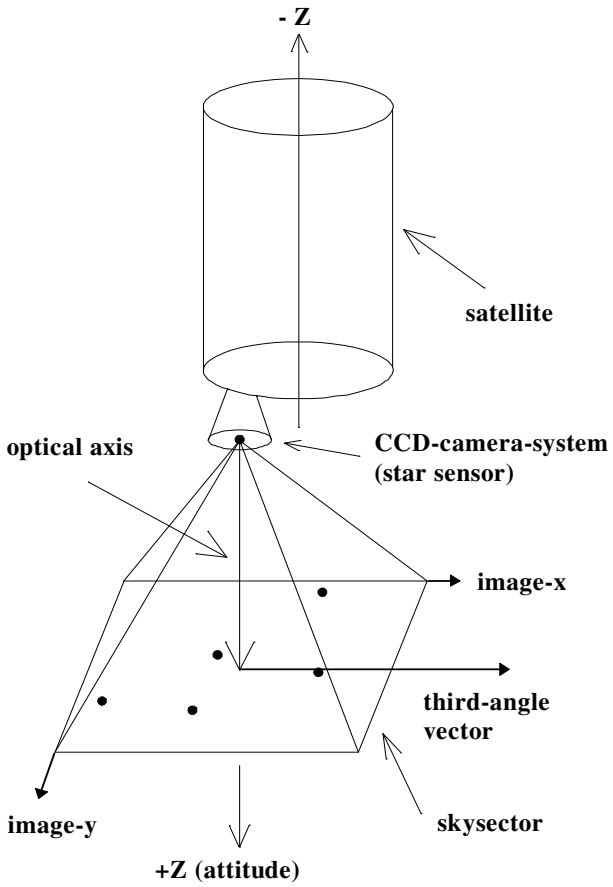

Fig. 2: Image of a skysector with different stars to be matched and attitude (+z) respectively *third angle* to be determinated

2. CONCEPT OF SOLUTION

The concept of solution reminds of a typical image processing system. A sequence of certain modules is executed and gives a good overview on all software components (fig. 3). The kernel of cause is the module *MATCHING*, which includes all the matching procedures. A vectorized star catalogue reduces the running time during *MATCHING*, because of using the vectorial scalar product to calculate the angle between the stars.

- **initialization variables / vectorization of star catalogue**
  - memory allocation / initialization of variables
  - transform celestial coordinates from 2-dimensional right ascension α / declination λ system to quasi 3-dimensional unit vector system [$\alpha, \delta \rightarrow$ x, y, z with: $x = \cos(\delta)*\cos(\alpha)$, $y = \cos(\delta)*\sin(\alpha)$, $z = \sin(\delta)$]

- **load camera data / read image file**
  - set-up camera (exposure time, sensitivity)
  - make exposure and get camera data
  - or read existing image file from PC-hard disk
  - transfer image data into program-internal image memory

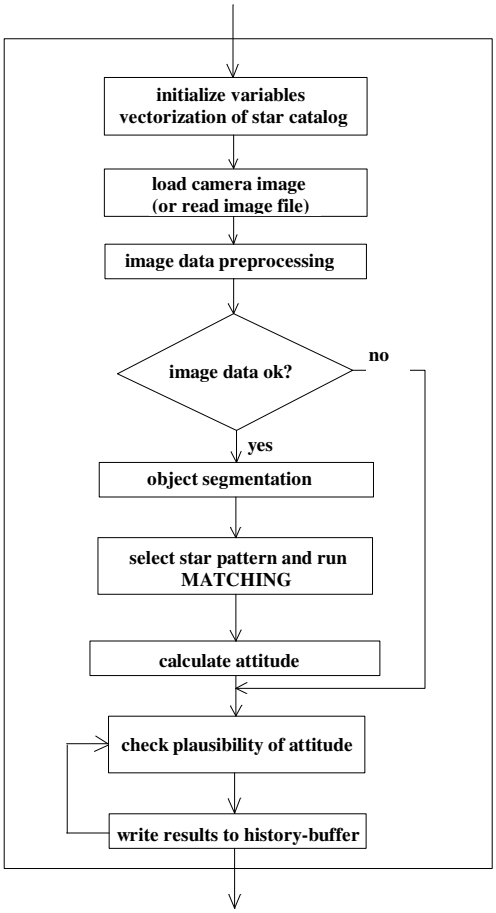

Fig. 3: Sequence of image processing modules

- **image data preprocessing**
  - delete pixel disturbances and non-stellar objects
  - calculate histogram, cumulated histogram and other statistical image parameters
  - check image parameter
  - decision of useful image

- **object segmentation**
  - get image-internal position (*x_hlp*, *y_hlp*) of all *high-level-pixels* (pixel brighter than given threshold)
  - segmentation of objects and store their area strong point positions (*x_obj*, *y_obj*) and gray values (*gray_value_obj*)
  - sort object data arrays in order to their gray values

- **select star pattern and run *MATCHING***
  - build star pattern of brightest stars (up to six stars) by using the gray value-sorted object list
  - calculate all the pixel distances and angle distances / init all matching variables
  - run *MATCHING* (see next chapter)

- **calculate attitude (orientation of body z-axis, *third angle*)**
  - determination of the 3-axis attitude, if star pattern was matched (three stars in minimum)

- **check plausibility of attitude**
  - get the best 3-axis attitude if a variety of pattern is matched by comparing the current results with former results via *history-buffer*

- **write results to history-buffer**
  - write the results and important parameters of the current image processing sequence to *history-buffer* (file on PC or EEPROM in spacecraft)

The star catalogues we use for matching are modifications of the well-known SAO-catalogue *(Smithsonian-Astrophysical-Observatory*). We reduce the SAO-catalogue to different sub-catalogues, to consider only these stars, which are detectable and bright enough for the CCD camera system. Planets are added automatically with their current positions in order to date and time. Manual editing allows handling potential errors of the star catalogues.

The job of the procedure *init-matching-variables* is to initialize some important matching variables (see also *Pascal* proc. 1). First the array *pixel-distance-xy* of all segmentated objects is multiplied by the so-called *angle-factor*. The *angle-factor* is the factor between the pixel distance on the CCD pixel array and the angle at the sky (degree per pixel). *Angle-factor* depends on the CCD pixel size and the focus length of the optic. The result is the *angle-distance-xy* array and we get a list of *angle-distance-min-xy* and *angle-distance-max-xy* by consideration of a tolerance called *angle-delta*. *Angle-delta* allows compensating inaccuracies of the star catalogue and little imaging failures of the optic. But *angle-delta* has a big influence on the behavior of the running, because so bigger *angle-delta*, so more star patterns are matched.. The last preparation before matching is running, is to calculate the cosine of the *angle-distance-min-xy* respectively *angle-distance-max-xy* array. At least we have a minimum allowed cosine and a maximum allowed cosine between all objects (arrays *of cos-xy-min* and *cos-xy-max*). Because of using the vectorial scalar product to calculate the celestial angle distances out of the star catalogue (*Pascal* func.1) it is enough to view only the cosine *cos-xy* of the current angles.

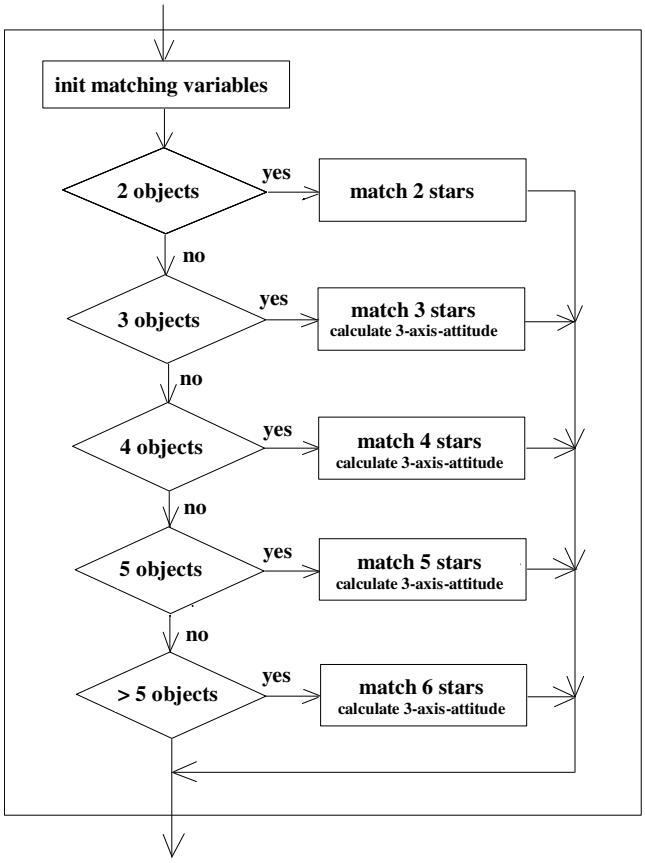

Fig. 4: Selectable matching procedures from two thru six objects

As shown in fig. 4 a minimum of two stars is needed for matching. The procedure *match-2-stars* tries to find all double stars, matching the given *pixel-distance-12* between the objects *1* and *2*. The procedure *match-3-stars* connects three given objects and tries to reconstruct this triangle (*pixel-distance-12*, *pixel-distance-13*, *pixel-distance-23*) via the star catalogue entries. At least the procedure *match-6-stars* looks for a pattern of six stars, if six stars are given in minimum. Within all matching procedures each object is connected to all other objects (2 objects → 1 link, 3 objects → 3 links, 4 objects → 6 links, 5 objects → 10 links, 6 objects → 15 links). For the additional 3-axis attitude determination a minimum of three matched stars is required.

To handle a potential variety of matching results during one image processing sequence we implemented a procedure *check-plausibility-of-attitude*. This procedure considers the results of former attitude determinations stored in the so-called *history-buffer*. For example: For spacecraft applications we can proceed from the assumption that a spacecraft with well-working attitude stabilization will not change the 2-axis attitude more than x degree within t minutes. The *history-buffer* is a ring buffer and contains data blocks with housekeeping-data like mission time or the most important parameters of former image processing sequences. The *history-buffer* is data-compressed and by down linking these informations from satellite to ground station the operator can comprehend the spacecrafts attitude behavior. Additional groundbased star pattern matching is easy possible by using the PC-software *MATCH-IT* and the down linked *history-buffer*. So the correction of onboard miss-matching during potential extreme situations is easy to handle via up linking the improved data.

3. MATCHING

The matching algorithm is explained on the example of a pattern matching with four stars. The *Pascal* procedure therefore is shown in proc. 2. Main loop (for w1...) and second loop (for w2...) check all possible double stars of the star catalogue. The star catalogue has *number_stars* entries. The double star (w1, w2) has to match the *pixel_distance_12* within the given image, respectively *cos_12* has to match the corresponding range between *cos_min_12* and *cos_max_12*. If the double star matching is successful, another loop (for w3...) is running to take the third star. The third star has to match the *pixel_distance_13* and *pixel_distance_23*, respectively the ranges of *cos_min_13* to *cos_max_13* and *cos_min 23* to *cos_max_23*. If this triple star matching is successful too, the inner and last loop (for w4...) checks a fourth star, whether it matches the connections to the three stars with the index w1, w2, and w3. If matching of the pattern of four stars was completely successful, we know which star is corresponding to the given objects within the image. Because the celestial coordinates of the stars and the x-y focal plane coordinates of the objects are known, it is possible to calculate the celestial coordinates of each point within the focal plane. Therefore simple vector algebra is needed. So it is easy to calculate the spacecrafts attitude especially if the optical axis of the CCD camera system points to the attitude to be determinated. At least we calculate the *third angle*. We define the *third angle* as the angle between the x-axis of the image (or the spacecrafts internal body x-axis) and the celestial equator.

```
function cos_dist(x1,y1,z1,x2,y2,z2:single):single;

{ calculates scalar-product - cosinus of angle between the 2 input star unit-vectors }

begin
  cos_dist:=x1*x2+y1*y2+z1*z2
end;
```

Func. 1: Vectorial scalar product to calculate cosine *cos_dist* between two vectorized star catalogue entries

```
procedure init_matching_variables;

begin
  angle_distance_12:=pixel_distance_12*angle_factor;
    .....
    .....
  angle_distance_34:=pixel_distance_34*angle_factor;

  cos_min_12:=cos(pi*(angle_distance_12-angle_delta)/180);
  cos_max_12:=cos(pi*(angle_distance_12+angle_delta)/180);
    .....
    .....
  cos_min_34:=cos(pi*(angle_distance_34-angle_delta)/180);
  cos_max_34:=cos(pi*(angle_distance_34+angle_delta)/180)
end;
```

Proc. 1: Initialization of the matching variables

```
procedure match_4_stars;

{ the vectorized star catalogue is stored via the pointer variables x^, y^ and z^ }
{ the star catalogue has number_stars entries                                      }

var
  w1,
  w2,
  w3,
  w4:word;

begin { procedure }

  init_matching_variables;
  get_matching_start_time; { get start_time to check computation time later }

  for w1:=1 to number_stars do { main-loop }
    for w2:=w1+1 to number_stars do { 2nd star }
      begin
        cos_12:=cos_dist(x^[w1],y^[w1],z^[w1],x^[w2],y^[w2],z^[w2]);
        if (cos_12<cos_min_12) and (cos_12>cos_max_12) then
          for w3:=1 to number_stars do { 3rd star }
            begin
              cos_13:=cos_dist(x^[w1],y^[w1],z^[w1],x^[w3],y^[w3],z^[w3]);
              cos_23:=cos_dist(x^[w2],y^[w2],z^[w2],x^[w3],y^[w3],z^[w3]);

              { object 1 --> star w1 and object 2 --> star w2 }
              if (cos_13<cos_min_13) and (cos_13>cos_max_13) and
                 (cos_23<cos_min_23) and (cos_23>cos_max_23) then
                for w4:=1 to number_stars do { 4th star }
                  begin
                    cos_14:=cos_dist(x^[w1],y^[w1],z^[w1],x^[w4],y^[w4],z^[w4]);
                    cos_24:=cos_dist(x^[w2],y^[w2],z^[w2],x^[w4],y^[w4],z^[w4]);
                    cos_34:=cos_dist(x^[w3],y^[w3],z^[w3],x^[w4],y^[w4],z^[w4]);
                    if (cos_14<cos_min_14) and (cos_14>cos_max_14) and
                       (cos_24<cos_min_24) and (cos_24>cos_max_24) and
                       (cos_34<cos_min_34) and (cos_34>cos_max_34) then
                      begin { MATCHED 4-STARS 1-2-3-4 }
                        calculate_attitude;
                        save_matching_parameter;
                        inc(matching_counter)
                      end { MATCHED 4-STARS 1-2-3-4 }
                  end; { 4th star }

              { object 1 --> star w2 and object 2 --> star w1 }
              if (cos_13<cos_min_23) and (cos_13>cos_max_23) and
                 (cos_23<cos_min_13) and (cos_23>cos_max_13) then
                for w4:=1 to number_stars do { 4th star }
                  begin
                    cos_14:=cos_dist(x^[w1],y^[w1],z^[w1],x^[w4],y^[w4],z^[w4]);
                    cos_24:=cos_dist(x^[w2],y^[w2],z^[w2],x^[w4],y^[w4],z^[w4]);
                    cos_34:=cos_dist(x^[w3],y^[w3],z^[w3],x^[w4],y^[w4],z^[w4]);
                    if (cos_14<cos_min_24) and (cos_14>cos_max_24) and
                       (cos_24<cos_min_14) and (cos_24>cos_max_14) and
                       (cos_34<cos_min_34) and (cos_34>cos_max_34) then
                      begin { MATCHED 4-STARS 2-1-3-4 }
                        calculate_attitude;
                        save_matching_parameter;
                        inc(matching_counter)
                      end { MATCHED 4-STARS 2-1-3-4 }
                  end { 4th star }
            end { 3rd star }
      end { 2nd star }
    end { main-loop };

  get_matching_end_time;        { get end_time to check computation time next }
  calculate_matching_total_time { total_time:=end_time-start_time             }
end; { procedure }
```

Proc. 2: Matching kernel for procedure *match-4-stars*

4. COMPUTER SET-UP

Only a standard personal computer (PC) equipped with floating-point-unit and 1Mbyte VGA-graphic device is required to run the software package called *MATCH-IT*. No additional hardware is needed for this low-cost groundbased system. The programming language for this program is *Borland-Pascal 7.0*; the operating system is *DOS*. For workstation and RISC-processor applications and the *UNIX* operating system we have an *ANSI-C* version available. In order to onboard computer applications during our former and current satellite projects there are also *Occam* and *Parallel C* versions existing, to support the transputer architecture of the *Inmos T8XX*-family.

The graphical desktop of *MATCH-IT* has a resolution of 1024 x 768 pixels and the user is guided comfortable by mouse click from one feature or menu to the next. The images which include the pattern of stars to be matched are readable from PC hard disk or by direct link to a special CCD camera system (see next chapter) via parallel interface. All examples below (fig. 7 to 10, tab. 2 and 3) are calculated on a PC with an *Intel-Pentium* 133 MHz processor and *Chaintech*-Mainboard at 66 MHz clock.

5. EXAMPLES

To show a well-known star pattern, an image of the constellation *Cassiopeia* is given (fig. 5 and 6). The exposure is taken with a small CCD camera system (star sensor) which we designed for spacecraft applications [3]. The camera provides a resolution of 384 x 288 pixels and has a field of view (fov) of about 20° depending on the focal length of the optic and the imaging area of the CCD sensor. This low-cost hardware is also designed to minimize noise and the influence of cosmic radiation. Active temperature control and variable imaging parameters like exposure time and sensitivity provide a universal capability, especially for autonomous onboard spacecraft applications. The exposure time of the groundbased image below (*Cassiopeia*, fig. 5 and 6) was 300 milliseconds and shows six stars listed in tab. 1 [4].

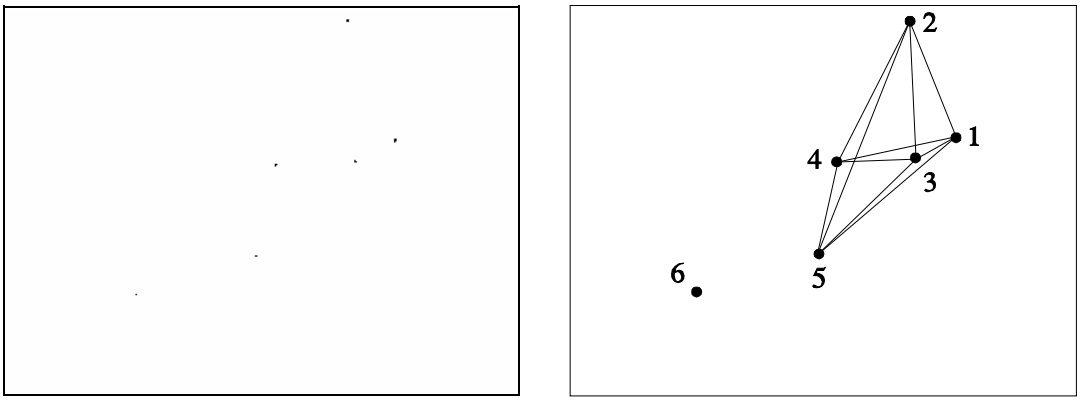

Fig. 5 and 6: CCD-snapshot of *Cassiopeia* (left) and distance-pattern between the five brightest stars (right)

star no. 1:	α Cassiopeia,	α: 0h 37.7m,	δ: +56°16′,	mag: 2.47	(Schedir)
star no. 2:	β Cassiopeia,	α: 0h 06.5m,	δ: +58°52′,	mag: 2.42	(Caph)
star no. 3:	η Cassiopeia,	α: 0h 46.1m,	δ: +57°33′,	mag: 3.64	(Achird)
star no. 4:	γ Cassiopeia,	α: 0h 53.7m,	δ: +60°27′,	mag: var	(Cih)
star no. 5:	δ Cassiopeia,	α: 1h 22.5m,	δ: +59°59′,	mag: 2.8	(Ksora)
star no. 6:	ε Cassiopeia,	α: 1h 50.8m,	δ: +63°25′,	mag: 3.44	(Segin)

Tab. 1: Stars within the constellation *Cassiopeia* respectively the images fig. 5 and fig. 6

A star catalogue including 1370 stars which covers the whole celestial globe with stars brighter than magnitude 4.9 was used to run matching and attitude determination procedures with 2, 3, 4, and 5 stars (fig. 7, 8, 9, and 10). The *angle-delta* is 0.25 degrees in this case. The figures 7 to 10 show the captured displays of *MATCH-IT*. The total computation time of this PC software is about 30% less than displayed within the parameter-box, because of the graphical work of the results. Matching with 3, 4 and 5 stars shows also the attitude determination by displaying the outer edges of the image and the *third angle* vector. More tabulated results on different matching parameters are shown in tab. 2 (*angle-delta*=0.25°) and tab. 3 (*angle-delta*=0.1°). The inner cells include the number of matches and the computation time (seconds, hundredth seconds).

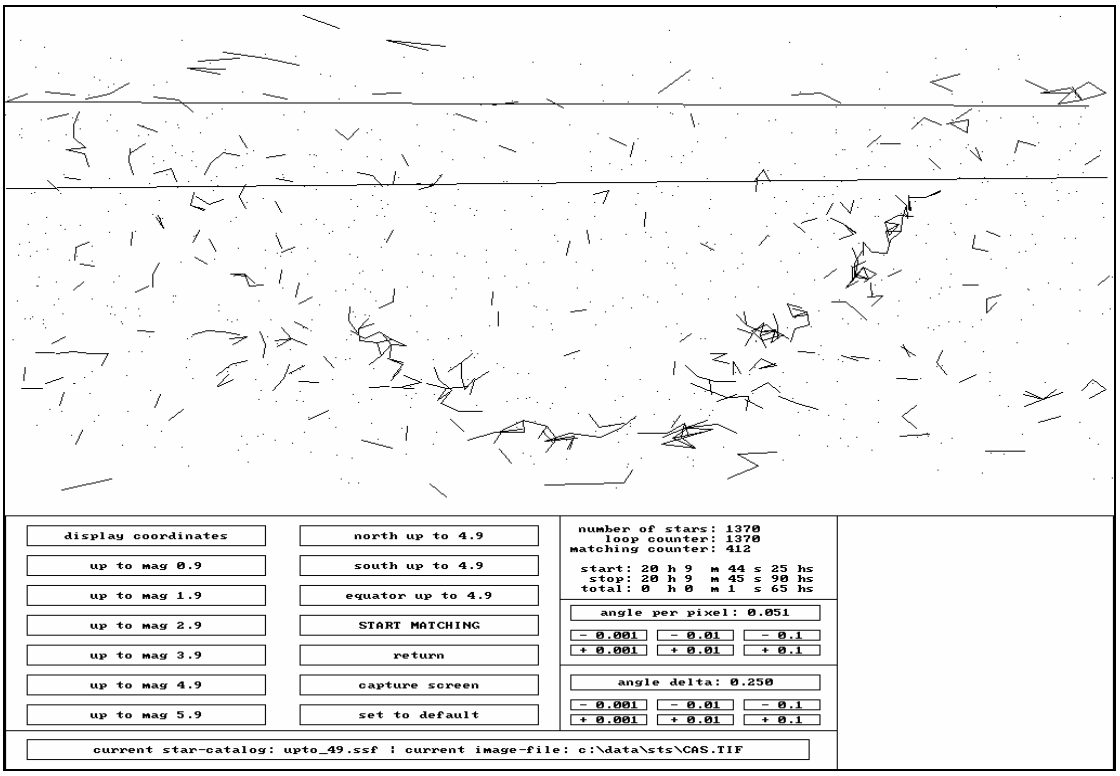

Fig. 7: Results of procedure *match-2-stars* on *Cassiopeia* (412 matches)

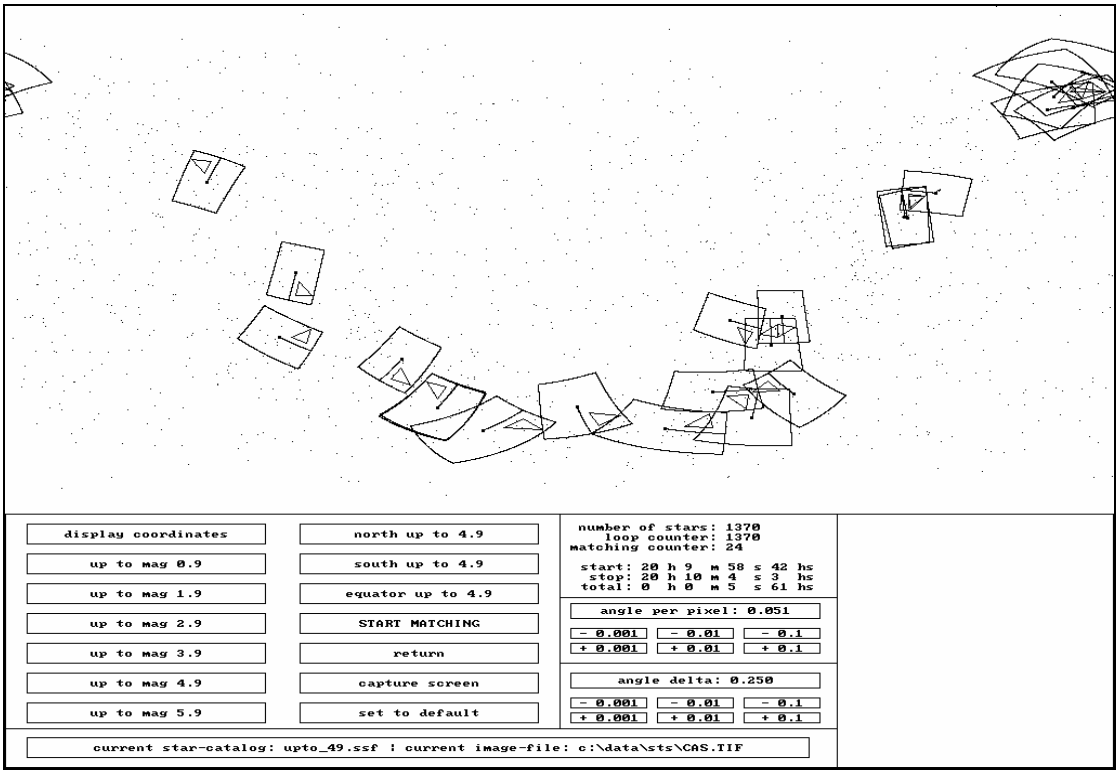

Fig. 8: Results of procedure *match-3-stars* on *Cassiopeia* (24 matches)

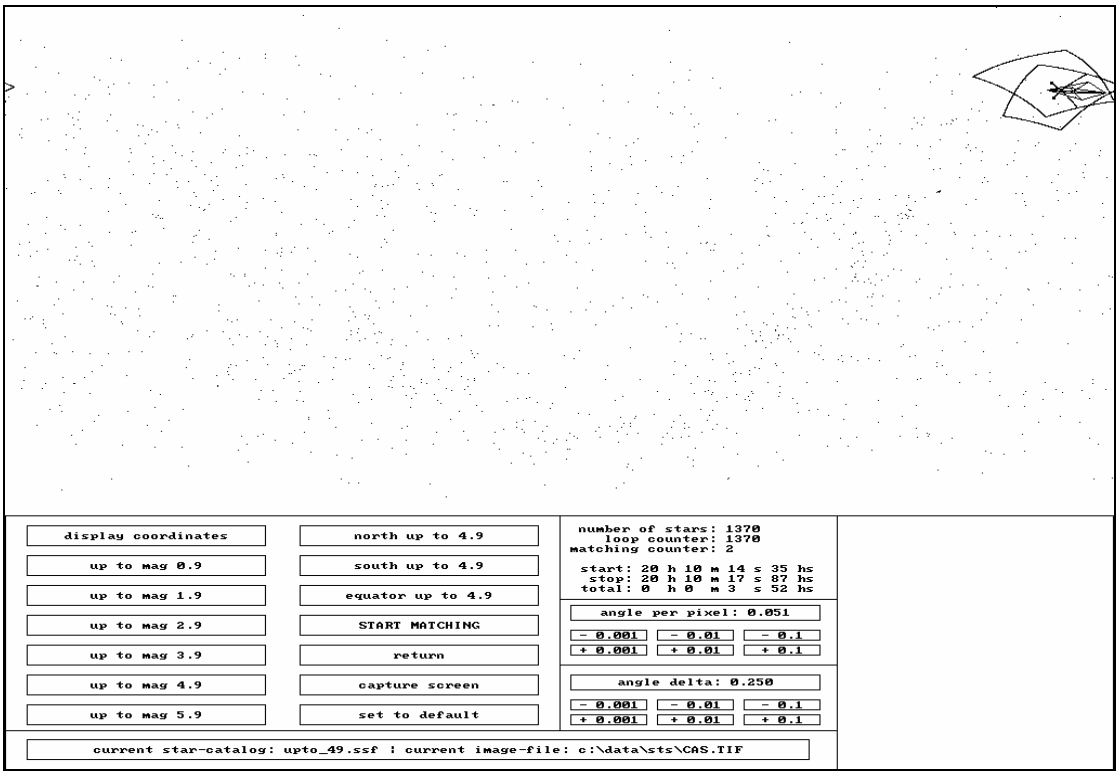

Fig. 9: Results of procedure *match-4-stars* on *Cassiopeia* (2 matches)

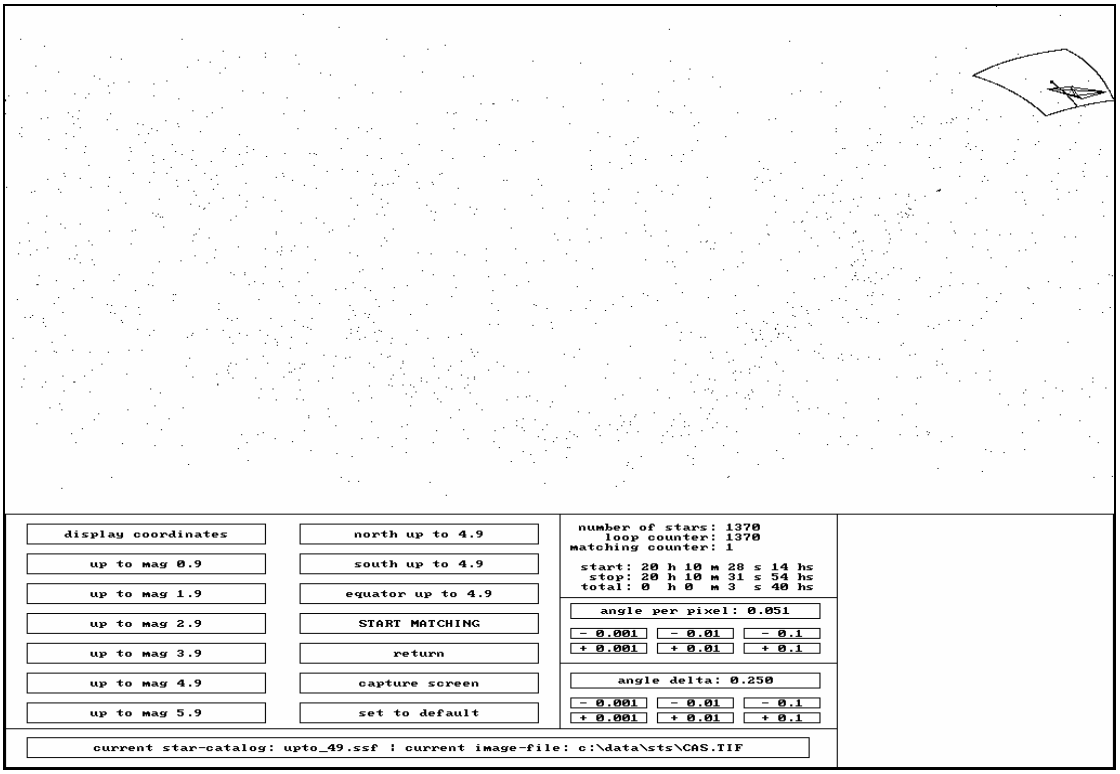

Fig. 10: Results of procedure *match-5-stars* on *Cassiopeia* (1 match)

	2 objects	3 objects	4 objects	5 objects	6 objects
12 stars (mag 0.9)	1 match < 0s 1hs	- < 0s 1hs	- < 0s 1hs	- < 0s 1hs	- < 0s 1hs
36 stars (mag 1.9)	2 matches < 0s 1hs	- < 0s 1hs	- < 0s 1hs	- < 0s 1hs	- < 0s 1hs
136 stars (mag 2.9)	9 matches < 0s 1hs	1 match < 0s 1hs	1 match < 0s 1hs	- < 0s 1hs	- < 0s 1hs
427 stars (mag 3.9)	32 matches 0s 16hs	1 match 0s 16hs	1 match 0s 17hs	1 match 0s 22hs	- 0s 22hs
1370 stars (mag 4.9)	408 matches 1s 64hs	24 matches 3s 19hs	2 matches 3s 30hs	1 match 3s 30hs	1 match 3s 35 hs
4480 stars (mag 5.9)	4117 matches 16s 80hs	790 matches 10s 58hs	41 matches 23s 38hs	3 matches 24s 26hs	1 match 25s 13hs

Tab. 2: Various matching results on *Cassiopeia* (fixed *angle-delta*=0.25°)

	2 objects	3 objects	4 objects	5 objects	6 objects
12 stars (mag 0.9)	- < 0s 1hs	- < 0s 1hs	- < 0s 1hs	- < 0s 1hs	- < 0s 1hs
36 stars (mag 1.9)	- < 0s 1hs	- < 0s 1hs	- < 0s 1hs	- < 0s 1hs	- < 0s 1hs
136 stars (mag 2.9)	4 matches < 0s 1hs	1 match < 0s 1hs	1 match < 0s 1hs	- < 0s 1hs	- < 0s 1hs
427 stars (mag 3.9)	13 matches 0s 16hs	1 match 0s 16hs	1 match 0s 16hs	1 match 0s 16hs	- 0s 17hs
1370 stars (mag 4.9)	146 matches 1s 53hs	1 matches 2s 9hs	1 matches 2s 9hs	1 match 2s 14hs	1 match 2s 14 hs
4480 stars (mag 5.9)	1624 matches 16s 4hs	58 matches 37s 13hs	3 matches 38s 6hs	1 match 38s 12hs	1 match 38s 50hs

Tab. 3: Various matching results on *Cassiopeia* (fixed *angle-delta*=0.1°)

6. CONCLUSION

We designed and implemented an image processing system to determinate star constellations and the attitude of spacecrafts as an application. A series of outdoor experiments was carried out to check and verify the capability of the whole system, beginning with the star sensor and ending with the 3-axis attitude determination. Software platforms are available for onboard computer applications in spacecrafts and for groundbased processing on standard personal computer. We did pay attention on the low-lost aspect, to serve especially the low-budget community like universities and small companies.

Now we are working to implement faster matching methods. Therefore binary trees are used to match the given stars within interlinked data arrays of angle distances.

7. ACKNOWLEDGEMENT

We thank our colleague Peter Offterdinger for support and taking the pictures with his self-built star sensor CCD camera system during rare clear nights at Bremen.

8. REFERENCES

[1] H. Renken: ''Design and implementation of an image processing system to detect star pattern'', *m.s.-thesis*, University of Bremen, FRG, 1992

[2] H. Königsmann, H. Renken, H. J. Rath: ''Attitude determination by image processing algorithms'', *6th AIAA/USU conference on small satellites*, Logan, Utah, USA, 1992

[3] P. Offterdinger, L. Rodehorst: ''Design and construction of a star sensor'', *m.s.-thesis*, Hochschule Bremen, FRG, 1993

[4] H. Vehrenberg, D. Blank: ''Handbook of the constellations'', Treugesell-Verlag, Düsseldorf, FRG, 1987