Ship Navigation Radar Tracking by Onshore Sensors for Coast Surveillance.

Robert Engebråten

Vestfold University College

Author Note

Robert Engebråten, Department of Maritime Studies, Vestfold University College.

This research was supported with counselling by the Norwegian Defence Research Department, who also lead the measurement campaign.

Correspondence concerning this master thesis should be addressed to Robert Engebråten, Hortensveien 252, 3157 Barkåker, Norway. E-mail: robert@ssdm.no.

Abstract

A proposed ship tracking system using the scan-based emitter passive localization technique has been analysed, based on real measurements. The stability of the radar scan-period and the variance of the measured sweep angles are investigated. An estimate on the systems geolocation accuracy and tracking range is presented based on these observations under the geographical coastal conditions in Finnmark, Norway. It was found that the proposed system in Finnmark has the potential of performing geolocation of ships with an accuracy of roughly 1000 m (CEP) at ranges up to 40 - 50 km from the coast, and 100 m at 20 km from the coast. In the same application, 14 sensors covered 285 km coastline.

Norwegian Defence Research Establishment (FFI) is investigating different ways for locating radars at sea, on shore, and in the air by measuring to the signals they emit. FFI employee Høye in 2010, wrote a report on geolocation with shipborn sensors by three different geolocation methods. In June 2012, FFI demonstrated that two sensors could track ships' positions (Teknisk Ukeblad, 2013). The measuring principle was based on *Time Difference of* Arrival (TDOA) and *Scan-Based Localization Technique.* The Norwegian inventor, Bjørn Hope, from the company Sensorteknikk AS in May 2012 described a system with a similar measurement setup, but based on the scan-based technique alone (Teknisk Ukeblad, 2012). Sensorteknikk has applied for patent on the system under the name *Distributed Ship Surveillance* (DSS), but the system's performance had yet to be documented. Sensorteknikk has applied for public founding from the Innovative Programme for Maritime Activities and Offshore Operations (MAROFF) proposed by the Research Council of Norway (Hope, 2013). In 2012 Sensorteknikk and FFI entered into cooperation (Teknisk Ukeblad, 2013). This master thesis is part of the planned R&D-activities as described in Sensorteknikk's application (Hope, 2013).

The aim of the thesis has been to investigate the performance of DSS. Will DSS provide acceptable ship geolocation accuracy, with a reasonably low sensor density along the coastline? And what range will it be able to track ship locations?

For this purpose a measurement campaign was initiated by FFI. Three laboratory prototype *Electronic Support* Measure sensors (ESM-sensors) where placed around the Oslofjord to record the radar activity in the area. The article presents observations on the stability of the radar scan-period and on the variance of the measured *Sweep Angles.* The geolocation accuracy and covering range of the DSS in a larger scale is estimated based on the observations from the Oslofjord and the geographical conditions at the coast of Finnmark.

Method

The theoretical analysis is based on scan-based localization technique. Ship positions obtained from live recordings are also solely based on scan-based localization technique. A basic understanding on the scan-based localization technique is necessary for the reader to relate to the arguments, hence a description of this technique will be introduced in the first subsection.

Scan-based localization technique

The scan-based localization technique is a technique to estimate the position of a scanning emitter (i.e. radar). It exploits the constant angular velocity of the antenna's mainlobe while it sweeps across receivers (sensors). The *swept angle* is the essential parameter for this technique. Figure 1 illustrates the geometric principle of attaining the swept angle α_g.

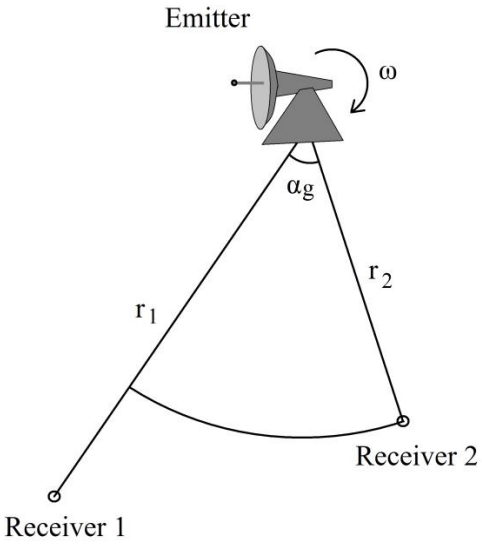

Figure 1. Model illustrating the geometric principle of attaining the swept angle α_g when the mainlobe of a scanning emitter, with angular velocity ω, sweeps over two receivers located at the distances r_1 and r_2 away from the emitter.

The mainlobe *time of arrival* (TOA) at receiver 1 and receiver 2 (Figure 1) is respectively denoted t_1 and t_2. Hmam (2007) argues that a planar model for determining emitter location is reasonable for applications where the emitter and receivers are located close to the earth's surface and the distances r_1 and r_2 are small enough to neglect the earth's curvature. Assuming that the emitted electromagnetic energy travels with the speed of light c, and that the emitter has a constant *scan-rate ω*, the *swept angle* α_g is given by:

$$\alpha_g = \omega(t_1 - t_2) - \omega\frac{r_1 - r_2}{c}. \quad (1)$$

The subscript *g* refers to the "geometric" angle as projected in the plane of the emitter vertex. Equation 1 can be rewritten as follows:

$$t_1 - t_2 = \frac{\alpha_g}{\omega} + \frac{r_1 - r_2}{c} + t_{com}. \quad (2)$$

t_{com} is added to account for the emitter scan-modelling and time measurement errors (Hmam, 2007). The second term in Equation 3 is due to uneven propagation delays of the electromagnetic energy. The distance difference $r_1 - r_2$ is by geometry always less than the baseline distance between the receivers. According to Hmam (2007), for most applications, a reasonable approximation of Equation 1 is obtained by neglecting the second and third term of Equation 2. A simplified equation for the swept angle can be written:

$$\alpha_g \approx \alpha = \omega(t_1 - t_2). \quad (3)$$

When the swept angle and receiver locations are known possible locations for the emitter can be drawn by moving the emitter in the plane between the receivers while maintaining the swept angle (Figure 2).

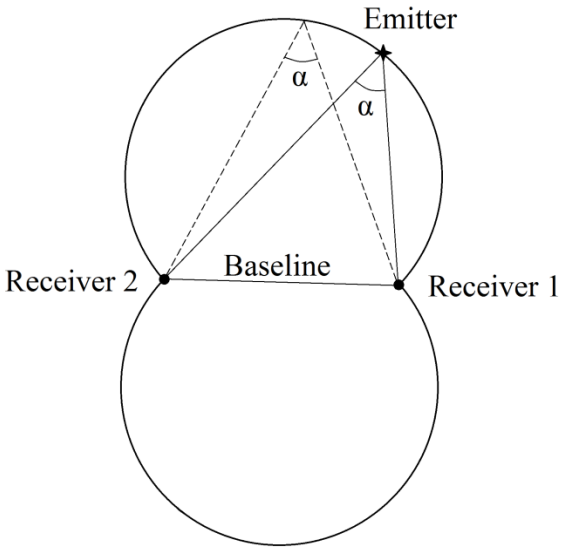

Figure 2. Possible emitter locations as function of the approximate swept angle α. The emitter may be located anywhere on the two circular arcs between the receivers.

An unambiguous solution for the emitter location cannot be determined by the scan-based localization technique with only two receivers. Other techniques for geolocating emitters are described in the litterature, e.g. Time Difference of Arrival (TDOA) (Kim, Song, & Musicki, 2012) and (Høye, 2010), *Angle of Arrival* (AOA) (Mikhalev & Ormondroyd, 2007) and (Høye, 2010), and *Frequency Difference of Arrival* (FDOA) (Mikhalev & Ormondroyd, 2007). But these techniques will not be considered in this thesis.

The scan-based localization technique requires at least three receivers for geolocation of emitters. Three receivers allow three different receiver-pair combinations and therefore give rise to three swept angles. The emitter is located at one of the four intersections of the two arcs as illustrated in Figure 3. Two solutions can be identified as the location of one of the receivers and therefore excluded. The remaining two solutions are the true emitter location in the sea sector and the false solution in the land sector.

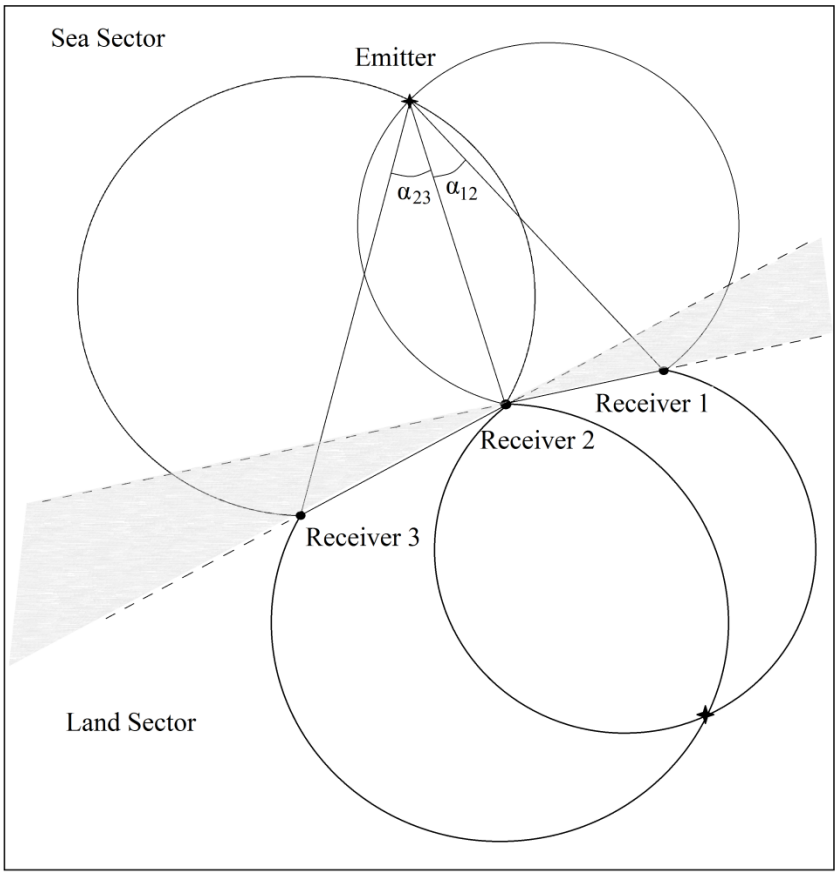

Figure 3. Intersecting arcs from the receiver combinations; Receiver 1 – Receiver 2 and Receiver 2 – Receiver 3. The two mirrored arcs intersects in three locations. One of the locations has the location of the receiver on the common arm of the two swept angles (in this case Receiver 2). Another false solution is found in the land sector, while the true emitter location is at the intersection in the sea sector.

Hmam (2007) describes how emitter locations can be derived for N receivers and how the approximate emitter location can be used to calculate a refined solution for the location to account for the neglected third term in Equation 2.

Radar Characteristics

The radar beam consists of a mainlobe with several sidelobes as illustrated by (Løvli, 2005) in Figure 4

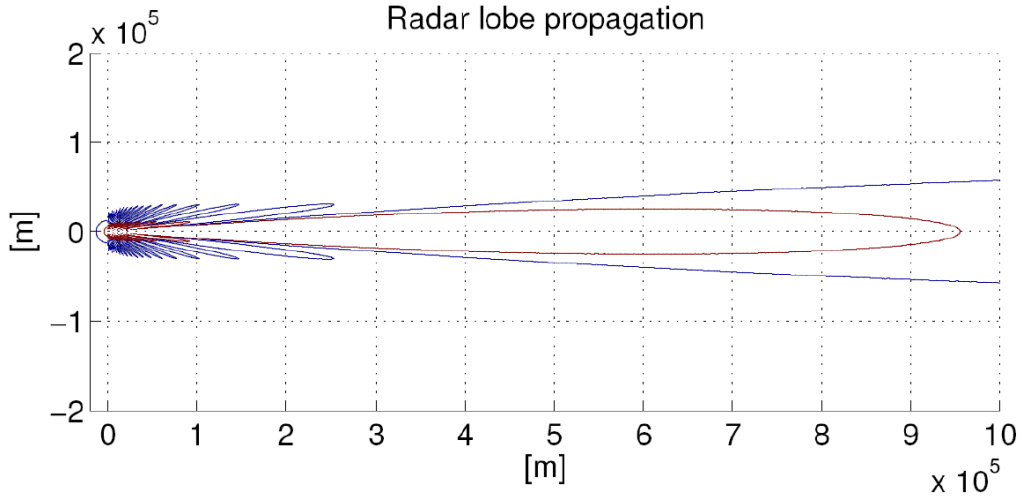

Figure 4. Radar lobe propagation pattern. Illustration borrowed from Løvli (2005). Radar is locatated at origo and the lobes emerges into the horizontal plane. The mainlobe is the middle red lobe whilst the blue lobes are sidelobes.

In vacuum, the radar beam travels with the speed of light in a constant direction. If we assume emitters and receivers to be located close to the earth's surface, the maximum distance between transmitter and receiver is limited by the curvature of the earth. In atmosphere, the *refractive index* changes and influences the direction of the beam. According to Barton (1988) there is no refraction relative to an enlarged imaginary earth with radius $R = \frac{4r}{3}$, where r is the actual earth radius. Figure 5 illustrates this.

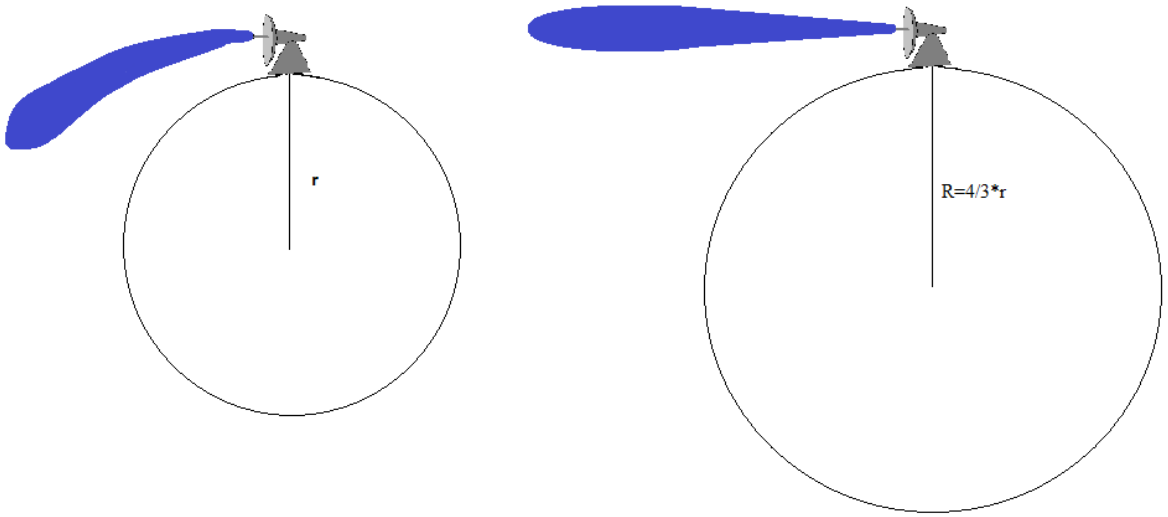

Figure 5. Refraction of the radar beam. The left circle illustrates the earth and show positive refraction of the radar beam. The right circle illustrates an enlarged imaginary earth with radius, R. Relative to the imaginary earth there is no refraction of the radar beam.

Radar Characterization

The DSS system is proposed to detect ships with unknown radar characteristics. The scan-based localization technique relies on a constant angular velocity. The radar's *pulse repetition interval* (PRI) and the emitted signals' *peak amplitude* are important in order to identify pulses emitted from the same radar. A measurement campaign was initiated to provide a set of *lobe description words*(LDW) containing data on the mainlobe's ToA, scan-period, amplitude and PRI.

Angular velocity.

The stability of the radar scan-period was investigated. The measurement campaign gave mainlobe data from three sensors containing scan-period for this purpose. The correlation between the measured scan-period where checked to determine if the instability was a result of measuring error or from actual fluctuations in the radar's scan-period. A stabile scan-period was used as an indicator on a stabile angular velocity.

Pulse repetition interval (PRI).

The time between emitted pulses is often constant. When it is so, PRI can help identify a series of mainlobes originating from the same radar. Typical radars in the Oslofjord have a PRI of 0.4 ms - 1.2 ms.

Accuracy Parameters

Cramer rao lower bound.

Cramer rao lower bound (CRLB) is a covariance matrix for a lower limit on how good an estimator can estimate a state vector based on a set of measurements. The theory of CRLB is found in statistical literature.

Circular error probable.

Circular error probable (CEP) is the radius of a circle, centred at the expected value of a random variable, containing 50% of the random variable outcomes. The theory of CEP is an often used accuracy parameter.

Measurement Setup

Three laboratory prototype electronic support measure (ESM) sensors where placed on strategic locations around the Oslofjord for detecting ships in the Horten area. The locations are shown in Table 1 and Figure 6a. The view from sensor 1 is shown in Figure 6b.

Table 1

Sensor Locations

	Coordinates	Location	Height
Sensor 1	[59.36906, 10.44221]	Vestfold University College	95 m.a.s.l
Sensor 2	[59.41545, 10.49698]	Norwegian Coastal Adm., Horten	10 m.a.s.l
Sensor 3	[59.53729, 10.53805]	Tofte, Hurum	44 m.a.s.l

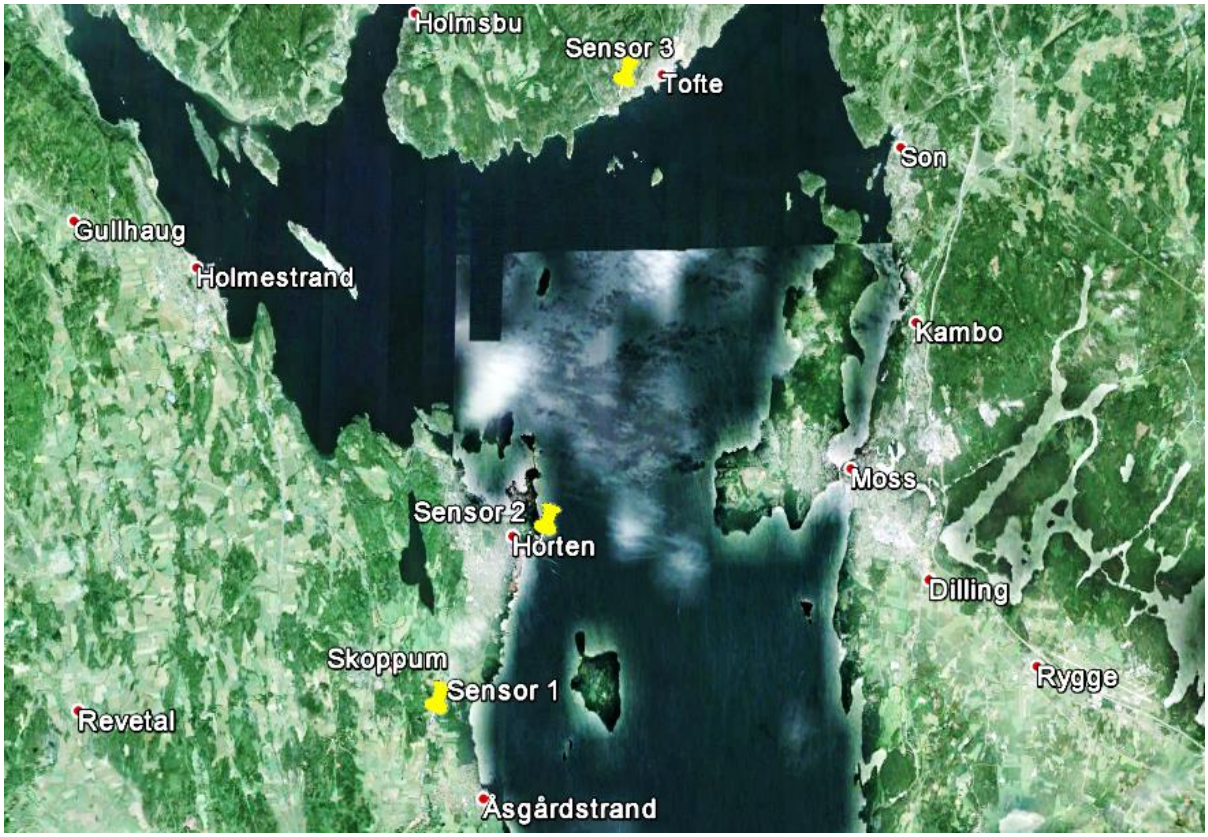

Figure 6a. Sensor Positions. (Created in Google Earth).

Figure 6b. Sensor 1 view. The sensor is weather protected inside a black plastic cylinder, taped to an existing mast on the roof of Vestfold University College. One of the Horten-Moss ferries is visible in the horizon.

In Figure 7, the signal flow of the pulses received at the sensors is illustrated. When a radar pulse hit the sensor the antenna is put into electric vibrations producing a signal. In the low noise front end device (LNFE) at the base of the antenna, the signal is amplified and mixed with the signal of a local oscillator, tuned by the tuning voltage V_t. The resulting intermediate frequency (IF) of 160 MHz (bandwidth typically ± 30 MHz) is transferred by coax cable to the "FFI ESM-processor" (the LINE-card in the bottom left corner) located inside an externally powered cabinet. The GPS device sets a timer in the FPGA every second. The timer starts a 100 MHz sampling, measuring the received signal every 10 ns. Whenever the signal exceeds the threshold voltage a process is started calculating, TOA, amplitude and pulse width which is written to a pulse description word (PDW) and transferred via USB connection to a local computer. In the computer, a LabView program checks for the 10 % strongest signals over a period of 0.1 to 1 seconds. If the amplitudes form a top, the top pulse is considered a mainlobe candidate, but it could also be a sidelobe. Then the next 10 % strongest signals are considered, etc. Three consecutive mainlobe candidates with same repetition interval and amplitude are considered "proof" of a mainlobe series. Grønvold and Oftebro in 2006 reported on a measurement campaign conducted in collaboration with FFI where strategies for recognizing mainlobes were discussed. Figure 8 is from Grønvold and Oftebro's report illustrating the problem with recognizing mainlobes. Detected mainlobes are written to lobe descriptive words (LDW) containing data on TOA, scan-period, PRI and amplitude for the strongest pulse. The LDWs are transfered to a central server via internett. Signal processing from measured signal to LDW was FFI's responsible in this measurement campaign. The central-processing pairs LDW data from sensor combinations and can be used to determine the radar location by the scan-based emitter passive localization technique. Analysis of the LDWs are described in the next section (Sweep Angles).

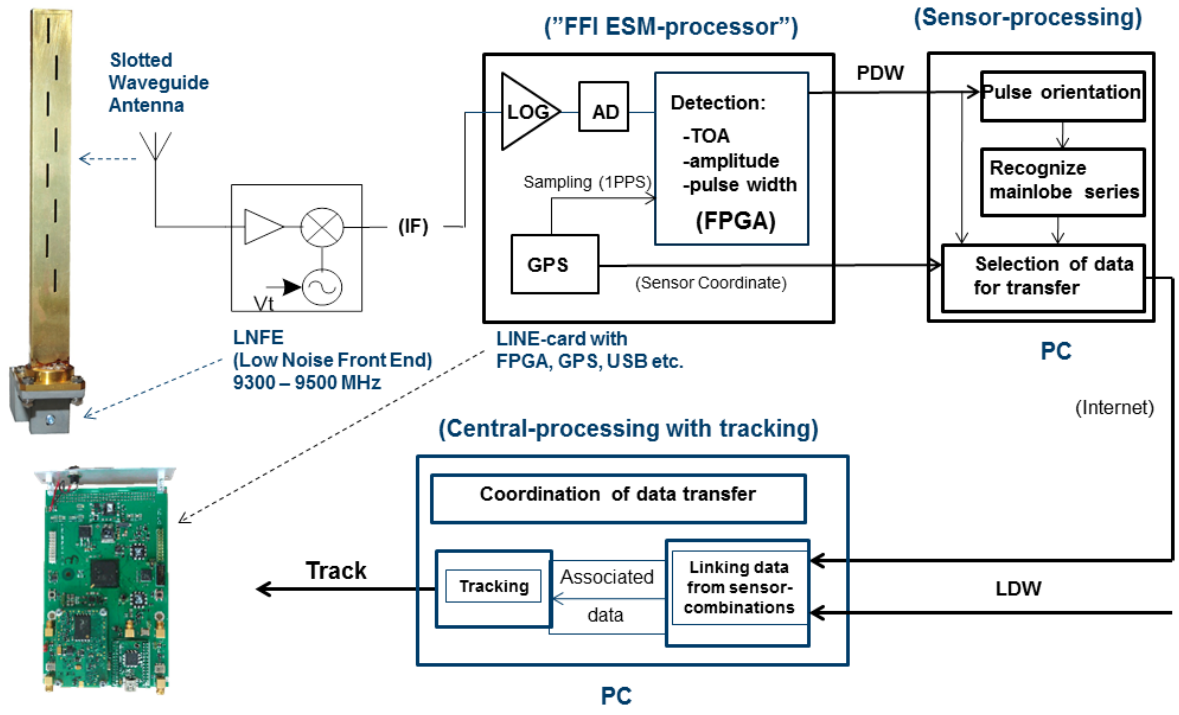

Figure 7. Adapted with permission from FFI (2011). Sensor arrangement with signal flow diagram.

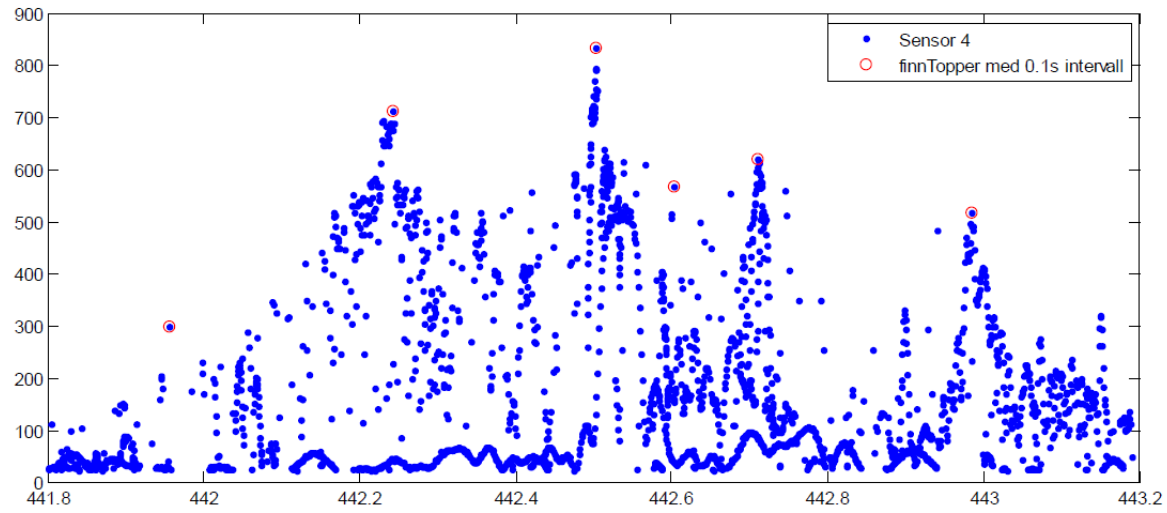

Figure 8. (Grønvold & Oftebro, 2006). Amplitudes written to PDWs from sensor 4 in Grønvold's and Oftebro's measurement campaign in 2006. The red circles mark the top of mainlobe candidates, determined over 0.1 second intervals. The figure show amplitudes recorded over 1.4 seconds, which is less than a typical radar scan-period. The 6 tops therefore probably originate from different radars.

Sweep Angles

The sweep angles were retrieved from the three sets of LDWs. The first step was to identify a radar that was simultaneously detected by all three sensors over time. By plotting PRI and scan-period data against mainlobe TOA (Figure 9 and Figure 10) a promising radar was found between 45 000 seconds and 48 000 second after midnight. PRI measured to be about 0.828 ms and scan-period was around 2.1 seconds. The data analysis is based on the recordings of this radar. The upper PRI limit was defined as 0.835 ms and the lower limit was set to 0.820 ms. All LDWs containing PRIs outside the limits were filtered out. See Attachment A for the complete code.

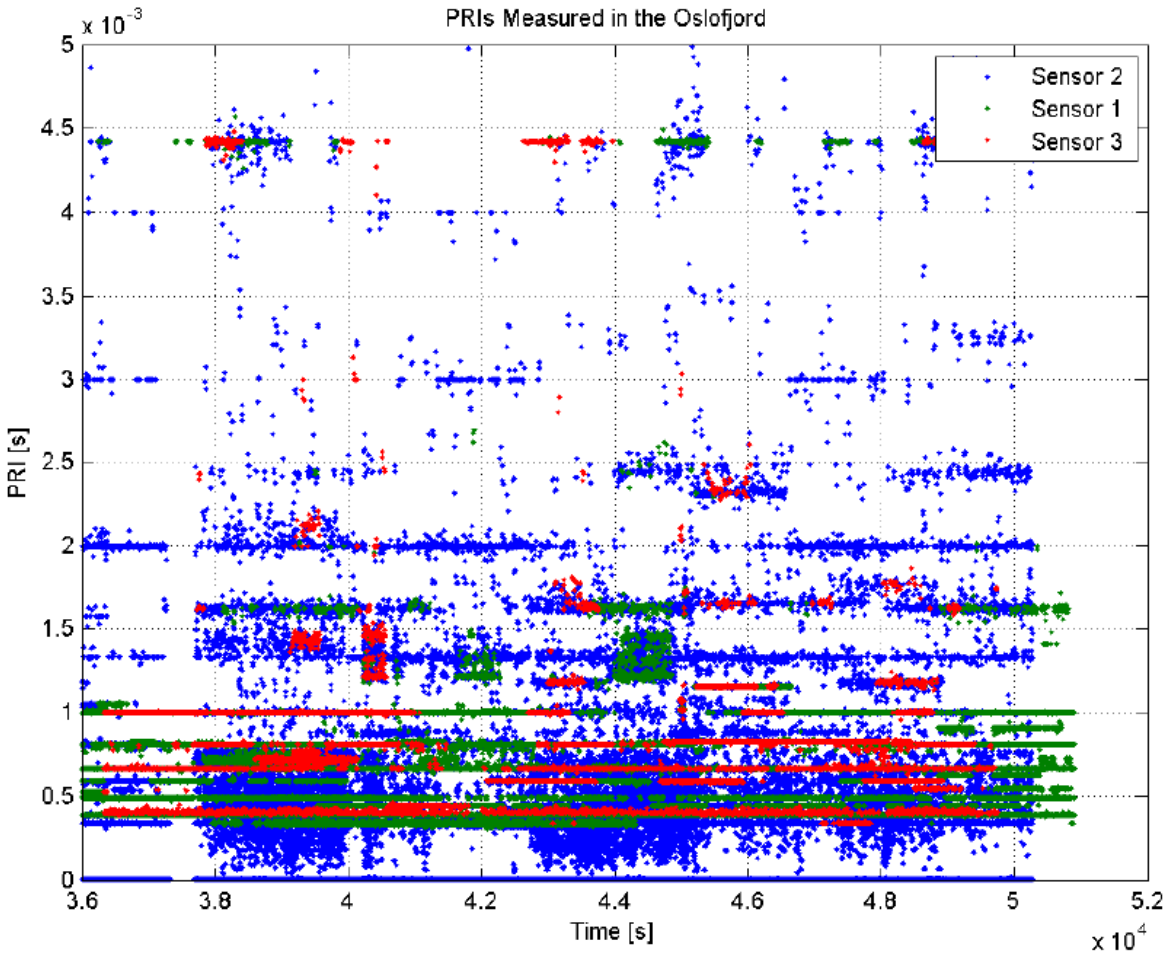

Figure 9. PRIs over 250 minutes (counting seconds from 00:00 am on recording day). PRI on all LDWs from all three sensors are plotted. The concentrated PRIs forming horizontal lines identify common radars.

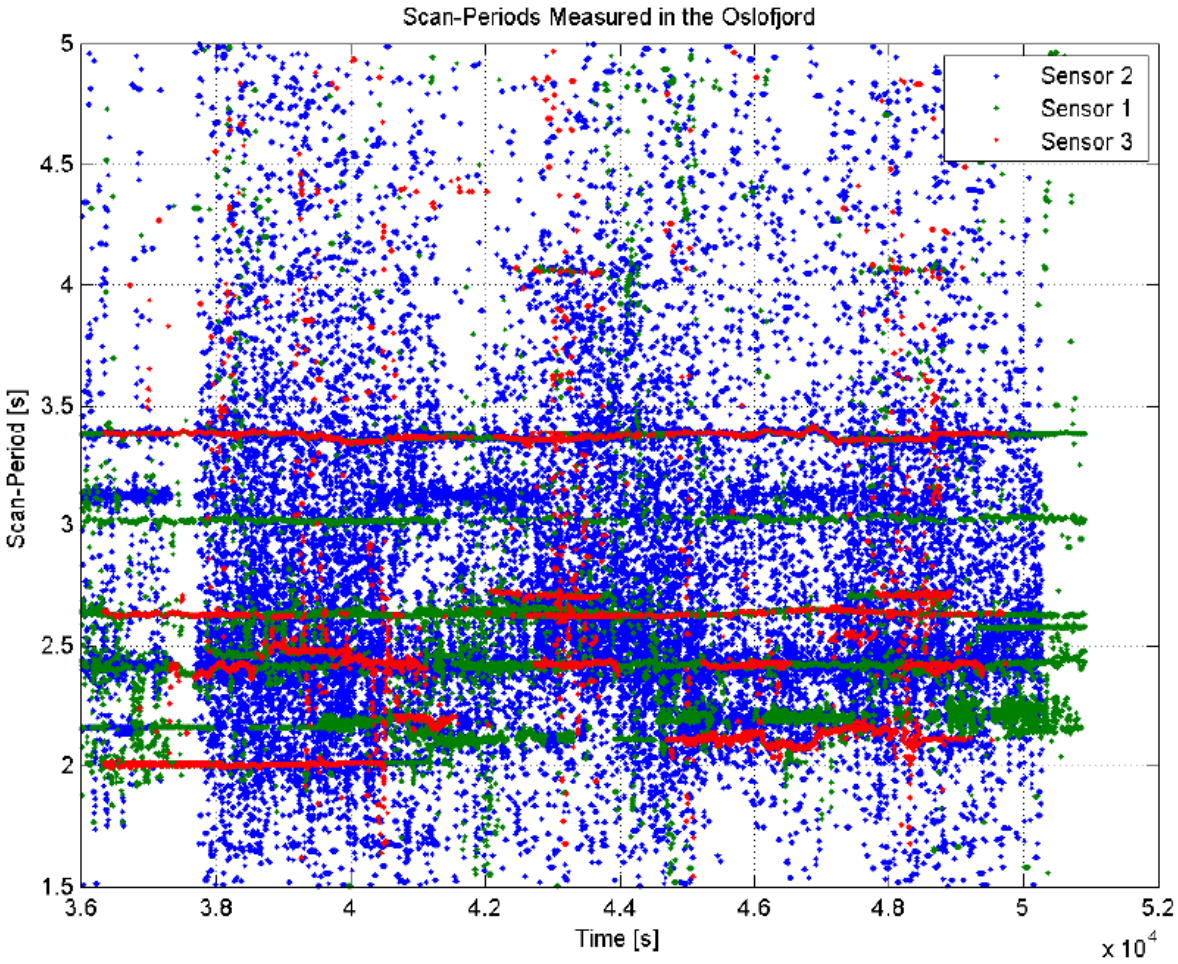

Figure 10. Scan-Periods over 250 minutes (counting seconds from 00:00 am on recording day). Scan-Periods on all LDWs from all three sensors are plotted. The concentrated scan-periods forming horizontal lines identify common radars.

Figure 11 show the scan-periods after PRI filtering. There are some aberrant scan-periods. To be sure that the remaining LDWs were all from the same radar LDWs containing aberrant scan-periods were also filtered out. Upper scan-period limit was set to 2.17. Lower limit was set to 2.06. After PRI and scan-period filtration it was revealed that the three sensors covered the subject radar simultaneously between 45 000 seconds and 48 000 seconds after midnight. The sweep angles were calculated from TOAs from the remaining LDWs. See Attachment A for the complete code.

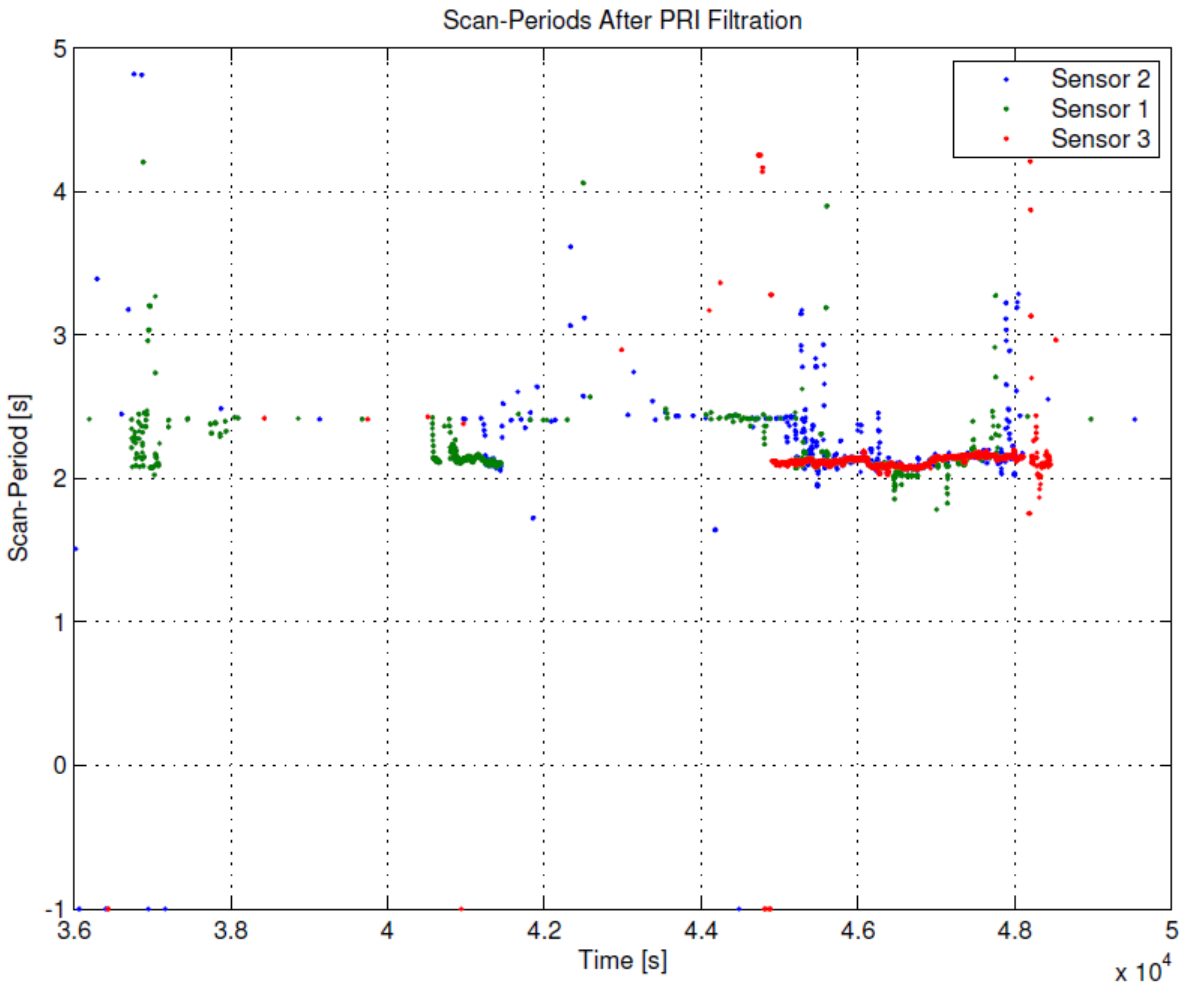

Figure 11. Scan-Periods After PRI Filtration. The diagram show scan-periods from LDWs with corresponding PRI between 0.820 – 0.835 ms on all three sensors.

A function of the subject radar scan-period was made by Matlab "shape-preserving interpolant" of the filtered scan-period. The scan-period function in Figure 12 was used in the algorithm (Attachment A) calculating swept angles.

10^{th} degree polynomial regression of the sweep angle sets were calculated:

$$E_{12}(x) = -0.14z^{10} + 0.014z^{9} + 1.1z^{8} + 0.1z^{7} - 3z^{6} - 0.67z^{5} + 3.9z^{4} \quad (4)$$
$$+0.76z^{3} - 2.6z^{2} + 3.7z + 170$$

$$\text{Where } z = \frac{x - 47000}{110} \tag{5}$$

$$E_{12}(x) = -0.12z^{10} - 0.21z^{9} + 0.8z^{8} + 1.3z^{7} - 1.6z^{6} - 2.5z^{5} + 0.6z^{4} + 1.7z^{3} + 0.45z^{2} + 2.1z + 120 \tag{6}$$

$$\text{Where } z = \frac{x - 47000}{80} \tag{7}$$

$$E_{13}(x) = -1.2z^{10} - 0.57z^{9} + 8.7z^{8} + 3.5z^{7} - 23z^{6} - 7.3z^{5} + 24z^{4} + 5.5z^{3} - 8.3z^{2} + 11z + 270 \tag{8}$$

$$\text{Where } z = \frac{x - 47000}{130} \tag{9}$$

It was assumed that Equation 4, 6 and 8 were almost equal to the expected angle. Hence the mean residual of the calculated angles compared with Equation 4, 5 and 6 was assumed a good estimate on the standard deviation of the calculated sweep angles. In other words:

$$E_{12}(x) \approx \mu_{12}(x) \qquad E_{23}(x) \approx \mu_{23}(x) \qquad E_{13}(x) \approx \mu_{13}(x) \qquad (10)(11)(12)$$

$$\sigma_{12} = \frac{\sum_{i=1}^{N}(\sqrt{(\theta_{12_i} - \mu_{12_i})^2})}{N} \tag{13}$$

$$\sigma_{23} = \frac{\sum_{i=1}^{N}(\sqrt{(\theta_{23_i} - \mu_{23_i})^2})}{N} \tag{14}$$

$$\sigma_{13} = \frac{\sum_{i=1}^{N}(\sqrt{(\theta_{13_i} - \mu_{13_i})^2})}{N} \quad (15)$$

Where θ denotes the calculated sweep angles and μ denotes the expected angle.

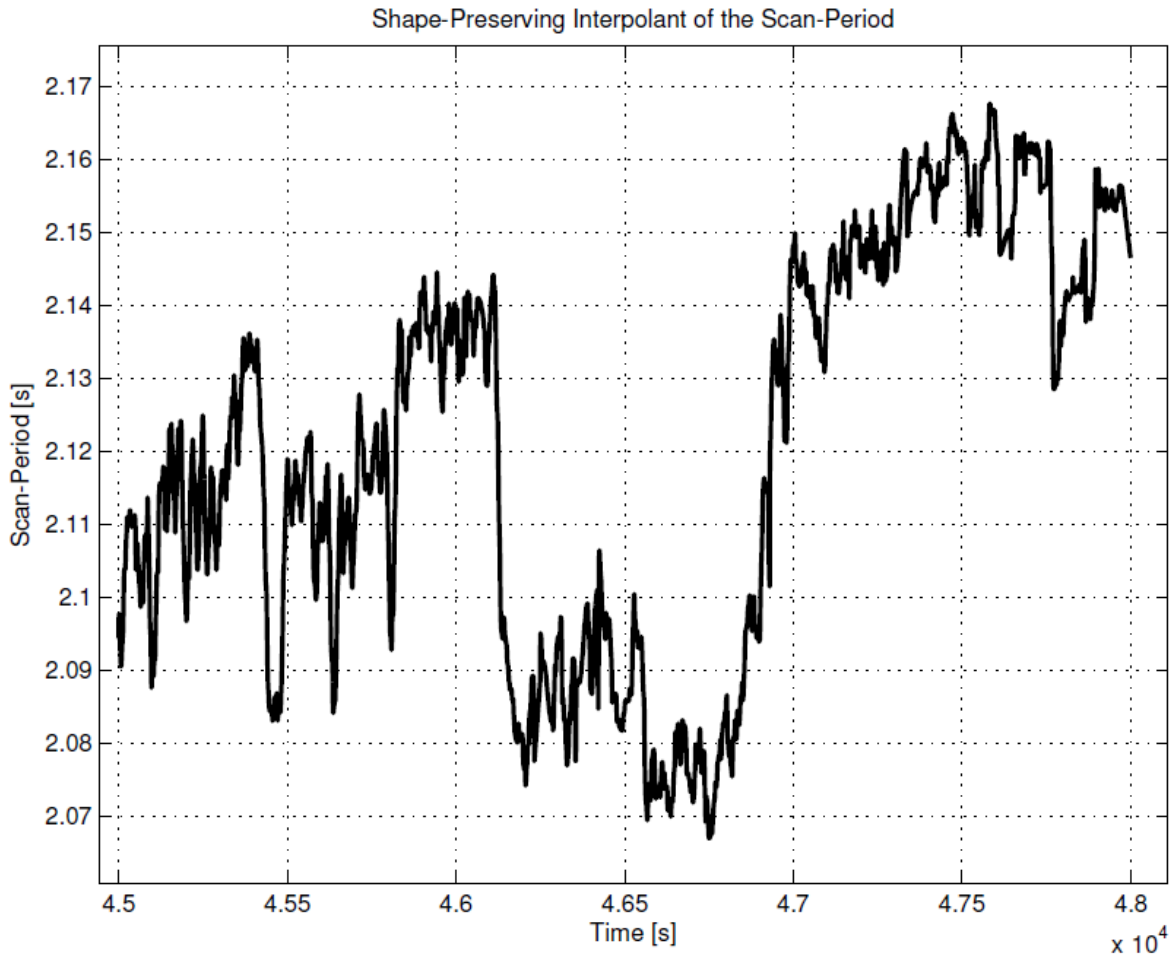

Figure 12. Matlab function, "shape-preserving interpolant" between 45 000 seconds and 48 000 seconds on scan-periods after filtration according to PRI and scan-period limits. This function was used as reference scan-period when calculating sweep angles.

Sensor Distribution Scale, Range and Geolocation Accuracy

A case study of the eastern Finnmark Coast in Norway, close to the Russian border, was conducted to estimate the performance of the DSS. The performance factors were; range, sensor distribution scale and geolocation accuracy.

Range.

Assumptions on how the refraction index' influence the radar beam were mentioned previously. The beam refraction affects the range of the DSS sensors. The range, or sight radius, is extended if the sensor or radar is elevated from the sea surface. For the analysis of the Finnmark coast, a radar elevation of 3 meters was considered. Assuming a circular earth with radius 6.359036×10^6 m, Figure 13 show the relationship between sensor height and sensor range. In the DSS each sensor covers a circular area. The coast area the DSS cover is therefore depending on the distance between the sensors. And because the DSS require TOA from three sensors to locate the radar, the covered coast area is limited even more (Figure 14).

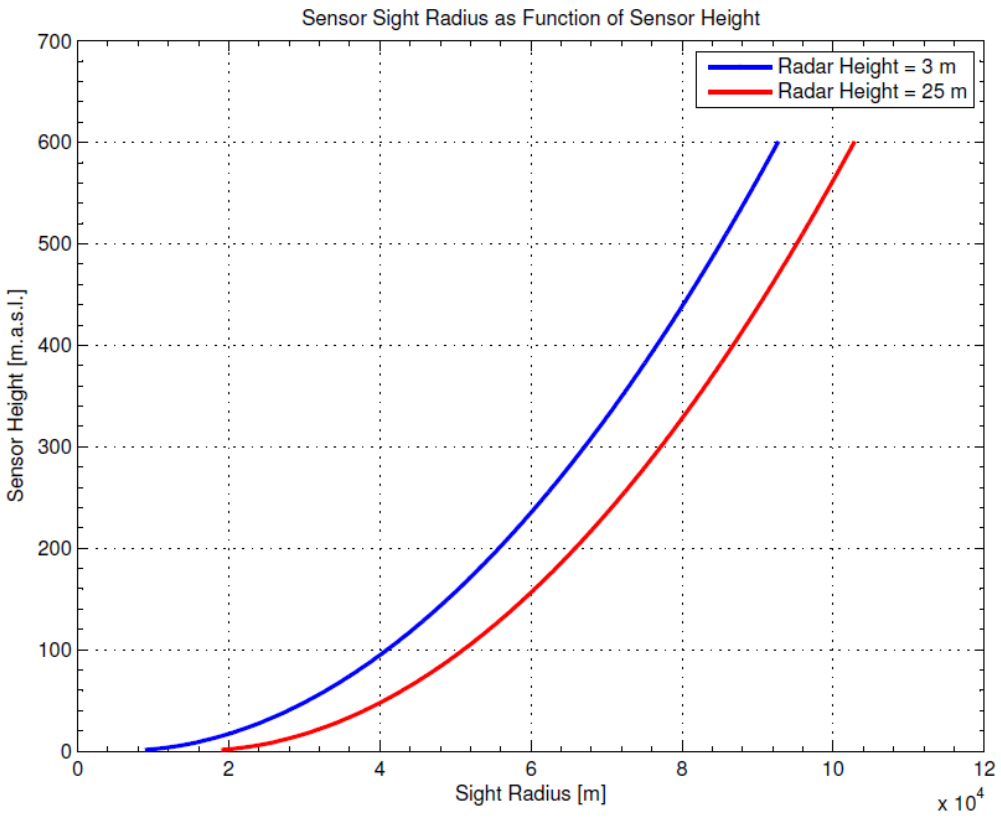

Figure 13. Sensor Sight Radius as Function of Sensor Height. Values on the horizontal axis mark the distance of the arc between the sensor location and the radar location on the earth surface, assuming a circular earth with radius corresponding to latitude 71°N. The highest elevated sensor in the Finnmark analysis was at 408 m.a.s.l., which according to this diagram would be able to detect radars 77 – 87 km away depending on the radar elevation.

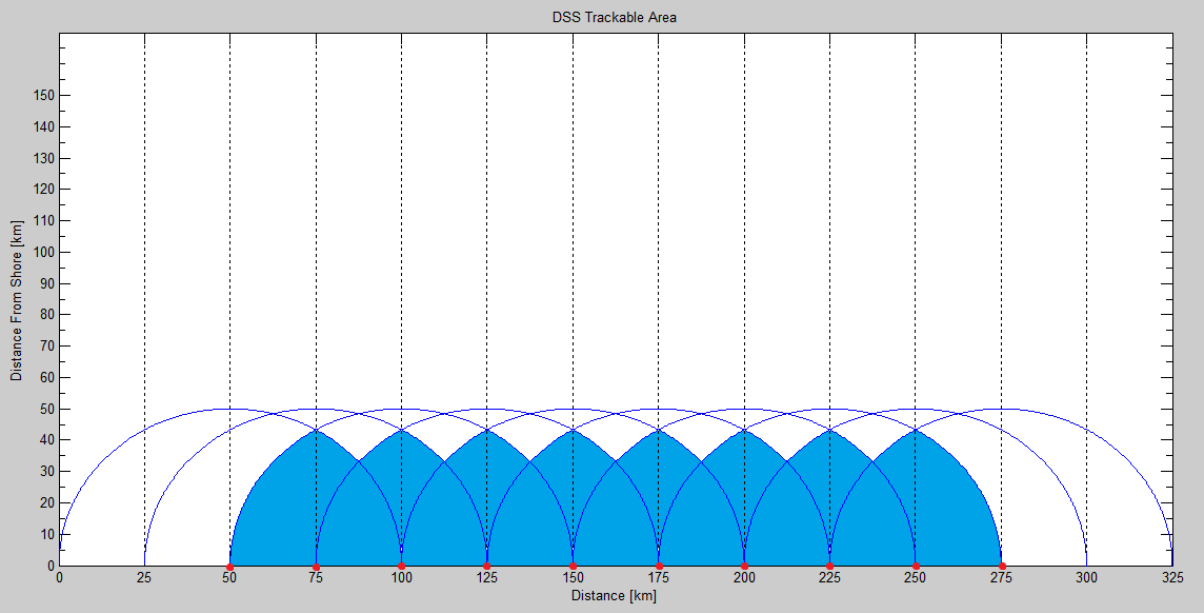

Figure 14. DSS Trackable Area. The diagram show sensors (red dots) with radius 2 times the distance between the sensors. The half circles illustrate each sensors sight radius. The blue area is covered by minimum three sensors, where continuous tracking of ships is possible with the scan-based emitter passive localization technique. Techniques where data from two sensors is sufficient for geolocation result in a greater trackable area.

Sensor distribution scale.

Sensors where distributed on the coast at the highest point close to the coastline with a good view of the sea. Ideal sensor locations where chosen by Google Earth searching. The distance between sensors where roughly set to half the average sight range of the two sensors, as in Figure 14.

Geolocation accuracy.

Standard deviations from the calculated sweep angles from the recordings of one of the radars in the Oslofjord were used for the case study of the Finnmark Coast. Based on the standard deviation estimates for the calculated sweep angles, standard deviation 0.45° was

chosen to calculate CEP-contours off the coast of Finnmark. The CEP-contours where plotted by the Matlab code in Attachment B (Smestad, 2013).

Results

Scan-Period Stability

Figure 15, 16 and 17 show the scan-period of one of the radars determined by data from respectively Sensor 1, Sensor 2 and Sensor 3. All three plots originate from the same radar and over the same time span. Figure 18 and Figure 19 illustrates the level of correlation for the scan-periods determined from data on the three sensors. Table 2 sum up the data statistics for the three scan-period sets. Table 3 and 4 quantifies the correlation of the scan-period sets.

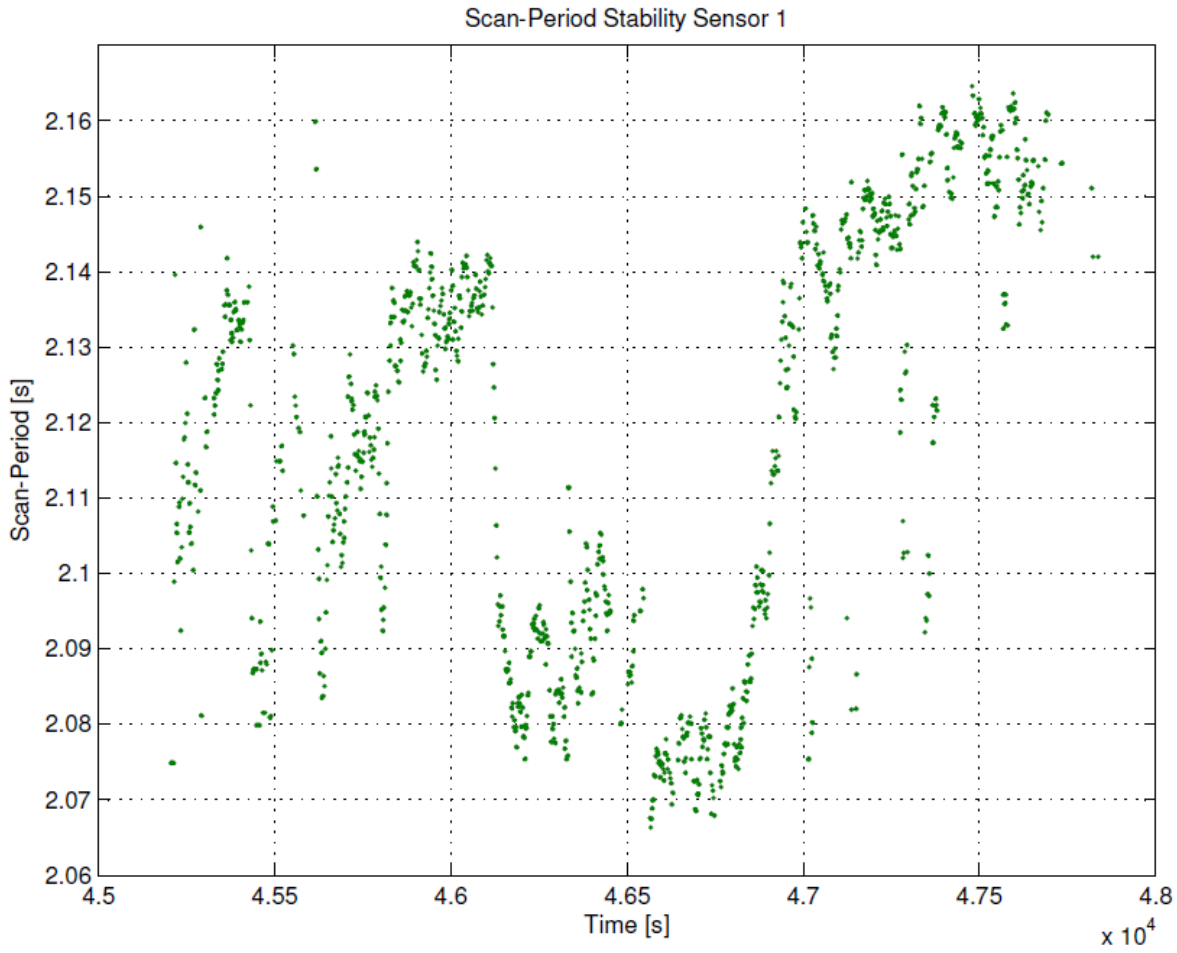

Figure 15. Scan-period trending over 50 minutes (45 000 - 48 000 seconds counting from 00:00 am on recording day). The scatter diagram shows scan-period as determined from Sensor 1 data.

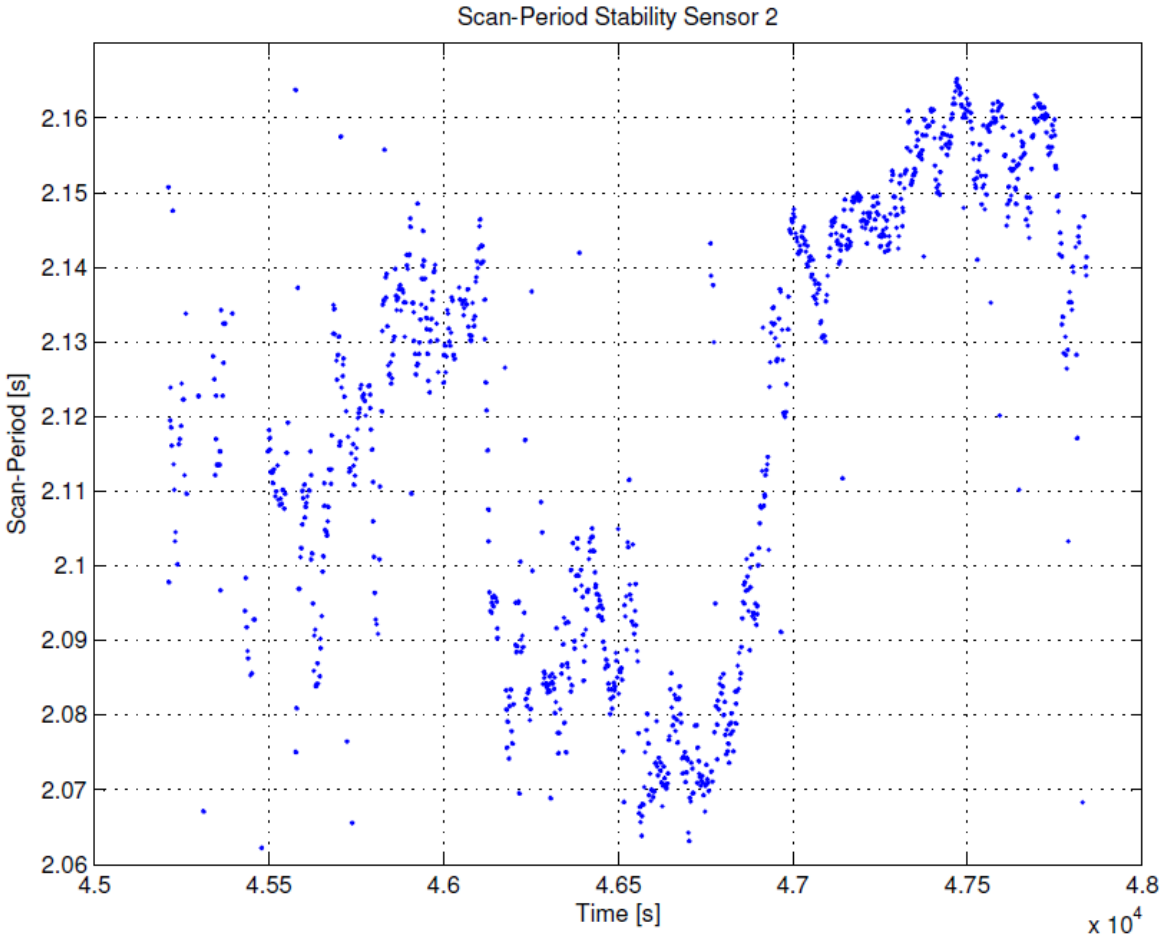

Figure 16. Scan-period trending over 50 minutes (45 000 - 48 000 seconds counting from 00:00 am on recording day). The scatter diagram shows scan-period as determined from Sensor 2 data.

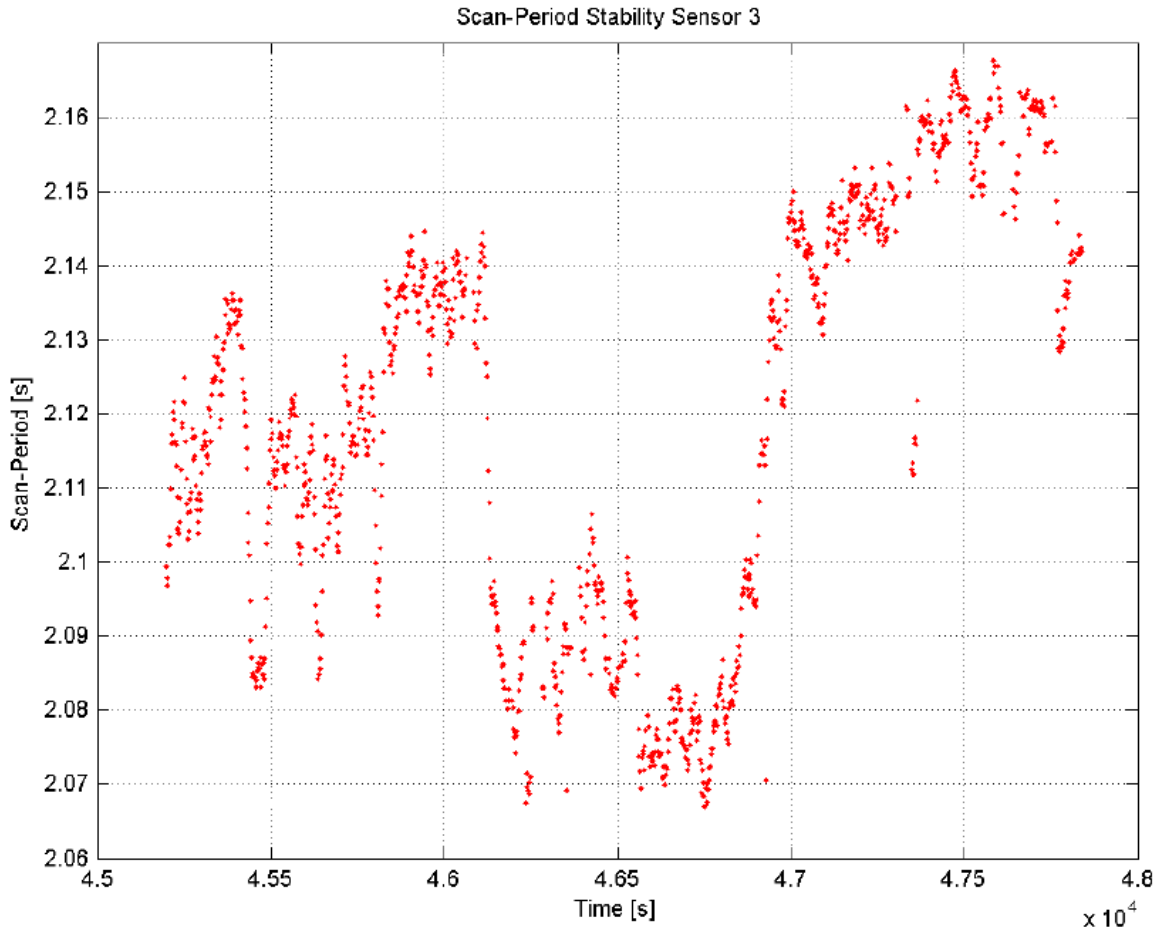

Figure 17. Scan-period trending over 50 minutes (45 000 - 48 000 seconds counting from 00:00 am on recording day). The scatter diagram shows scan-period as determined from Sensor 3 data.

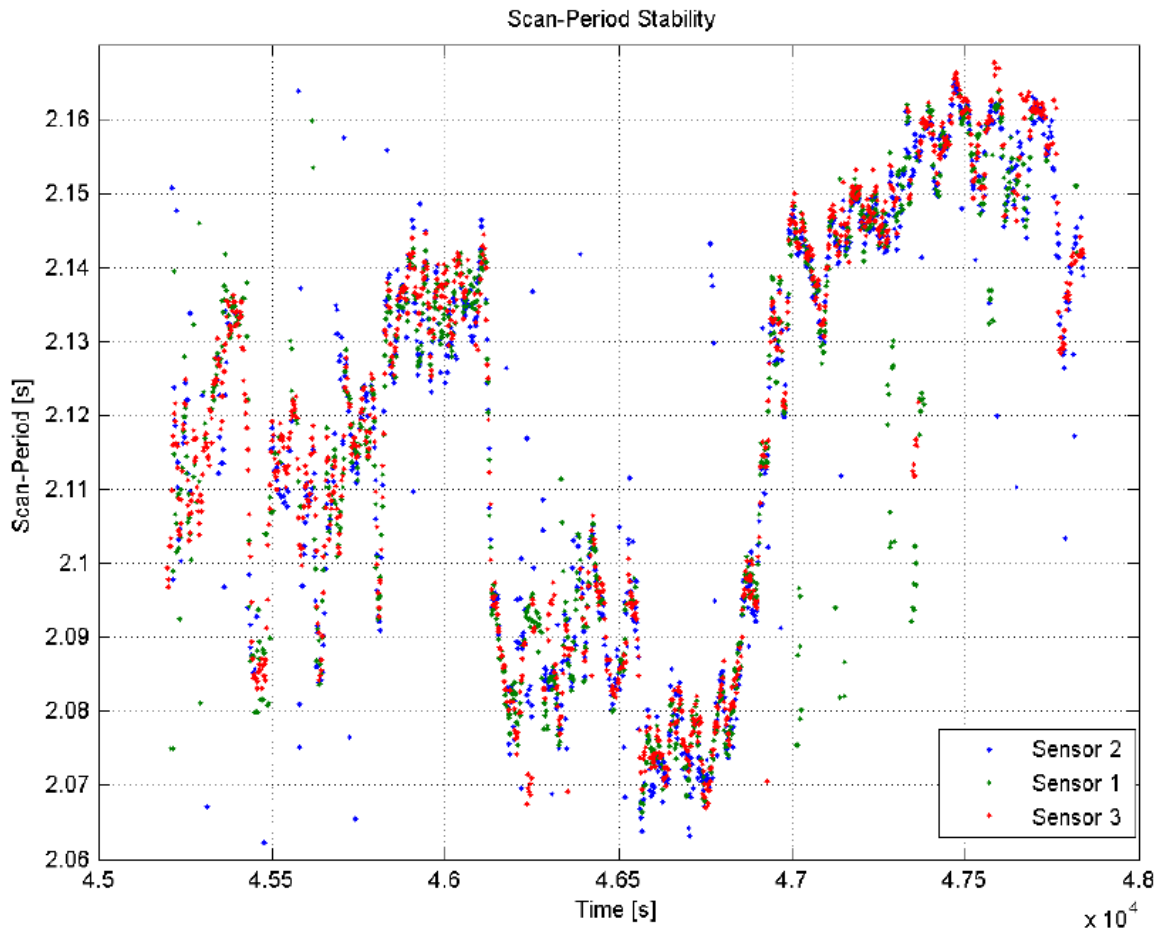

Figure 18. Correlation diagram over 50 minutes (45 000 - 48 000 seconds counting from 00:00 am on recording day). The scatter diagram illustrates correlation for scan-period as determined by Sensor 1, Sensor 2 and Sensor 3.

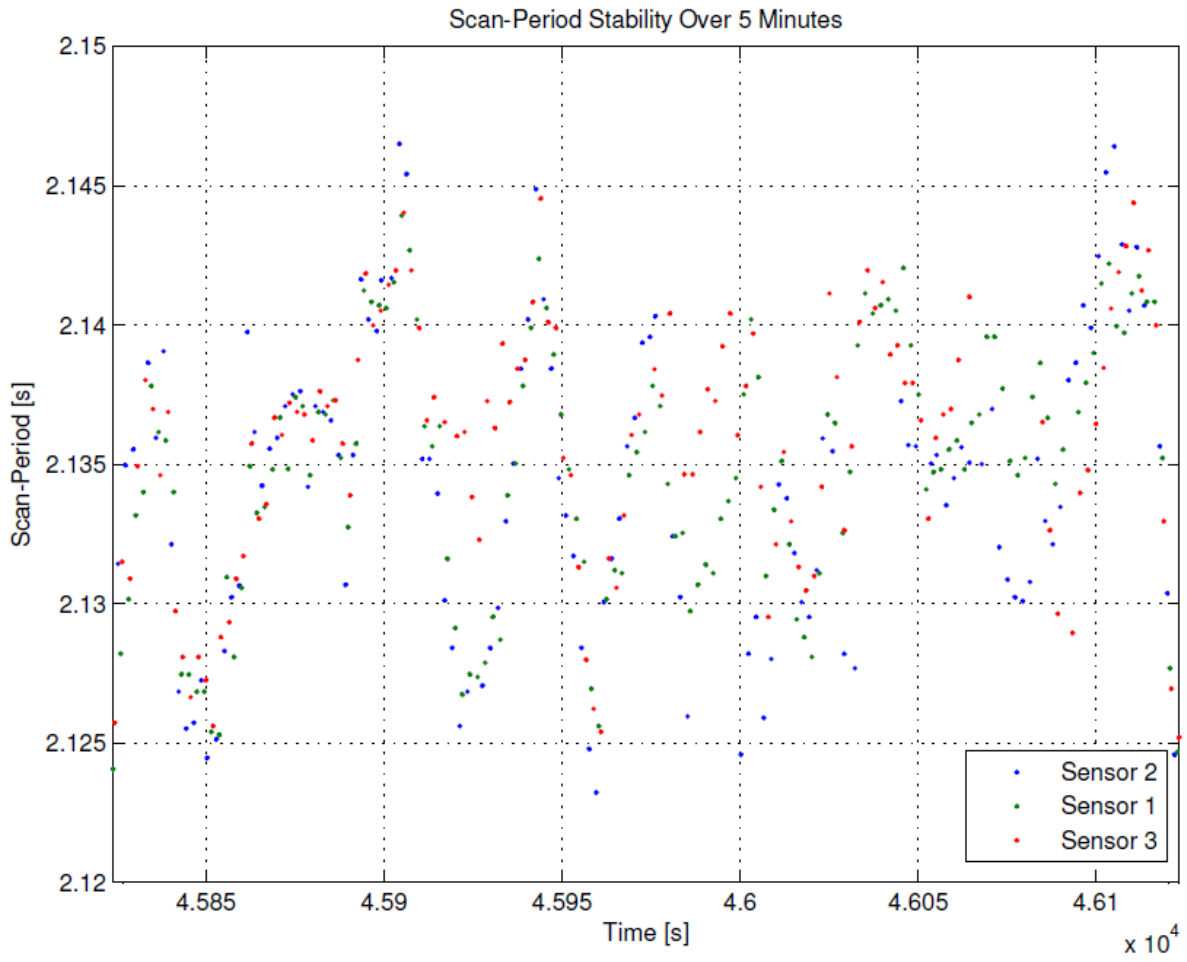

Figure 19. Correlation diagram over 5 minutes (counting seconds from 00:00 am on recording day). The scatter diagram illustrates correlation for scan-period as determined by Sensor 1, Sensor 2 and Sensor 3. Time between grid lines on the time axis is 50 seconds. Over 50 seconds scan-periods for 23 mainlobes per sensor are detected. Averaging over 5 mainlobes, corresponding to about 11 seconds, seems appropriate.

Table 2

Data Statistics for Scan-Period

	50 Minutes			5 Minutes		
	Sensor 1	Sensor 2	Sensor 3	Sensor 1	Sensor 2	Sensor 3
Min.	2.066	2.062	2.067	2.124	2.123	2.125
Max.	2.165	2.165	2.168	2.144	2.147	2.145
Mean	2.117	2.12	2.119	2.135	2.134	2.136
Median	2.121	2.127	2.12	2.135	2.135	2.137
Mode	2.075	2.161	2.072	2.131	2.128	2.128
Std	0.0276	0.0293	0.0279	0.00461	0.00535	0.00451
Std%	1.30 %	1.38 %	1.32 %	0.22 %	0.25 %	0.21 %

Table 3

Correlation for Measured Scan-Periods over 50 minutes

	Sensor 1	Sensor 2	Sensor 3
Sensor 1	1.0000	0.8898	0.9333
Sensor 2	0.8898	1.0000	0.9453
Sensor 3	0.9333	0.9453	1.0000

Table 4

Correlation for Measured Scan-Periods over 5 minutes

	Sensor 1	Sensor 2	Sensor 3
Sensor 1	1.0000	0.6515	0.8305
Sensor 2	0.6515	1.0000	0.6593
Sensor 3	0.8305	0.6593	1.0000

Sweep Angles

The calculated sweep angles from the subject radar's measured TOAs are shown in Figure 20, 21 and 22. The 10th degree polynomial regression of limited time ranges of these angle sets are plotted along with their residuals in Figure 23, 24 and 25. Table 5 show the estimate on sweep angle standard deviation.

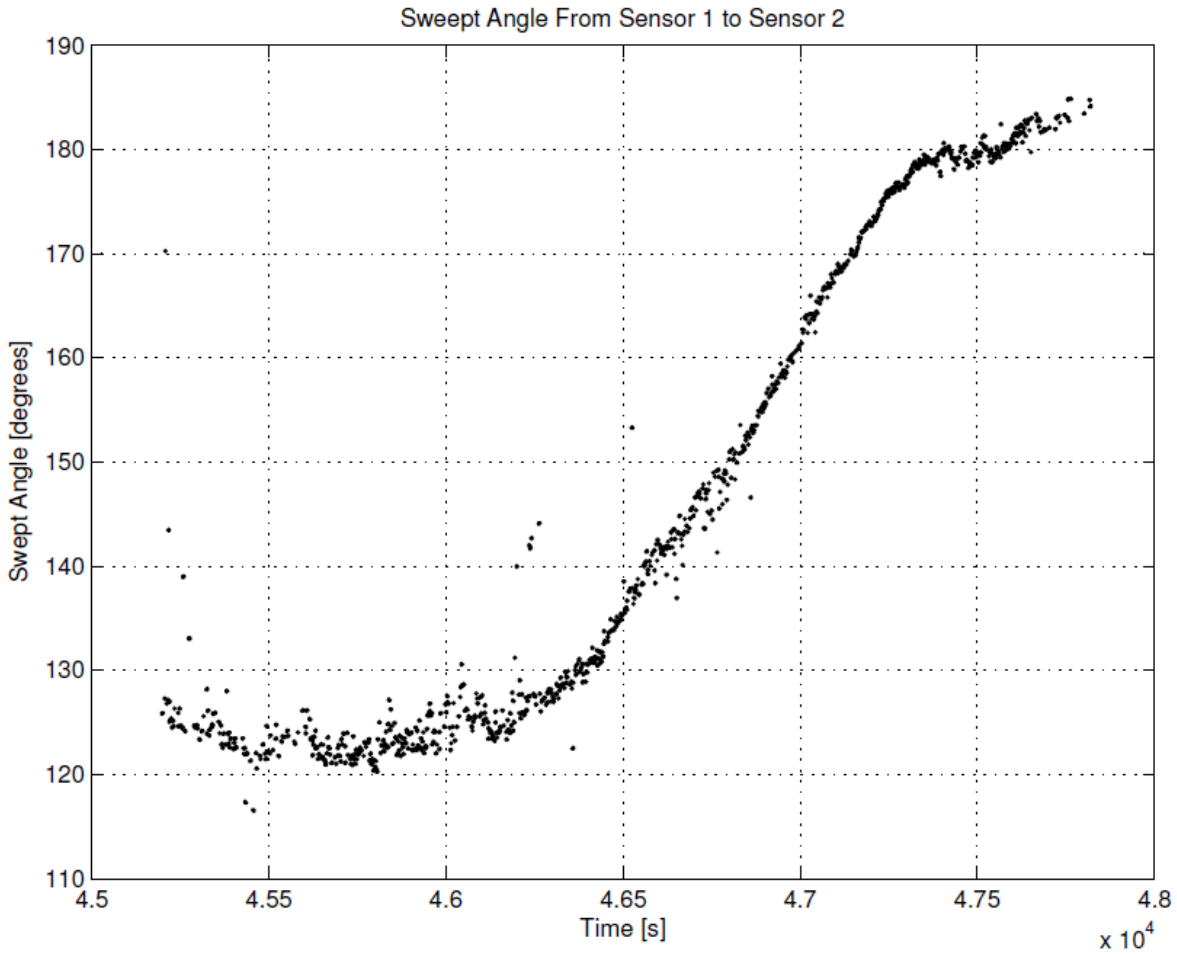

Figure 20. Calculated sweep angles between sensor 1 and sensor 2 over 50 minutes (45 000 - 48 000 seconds counting from 00:00 am on recording day).

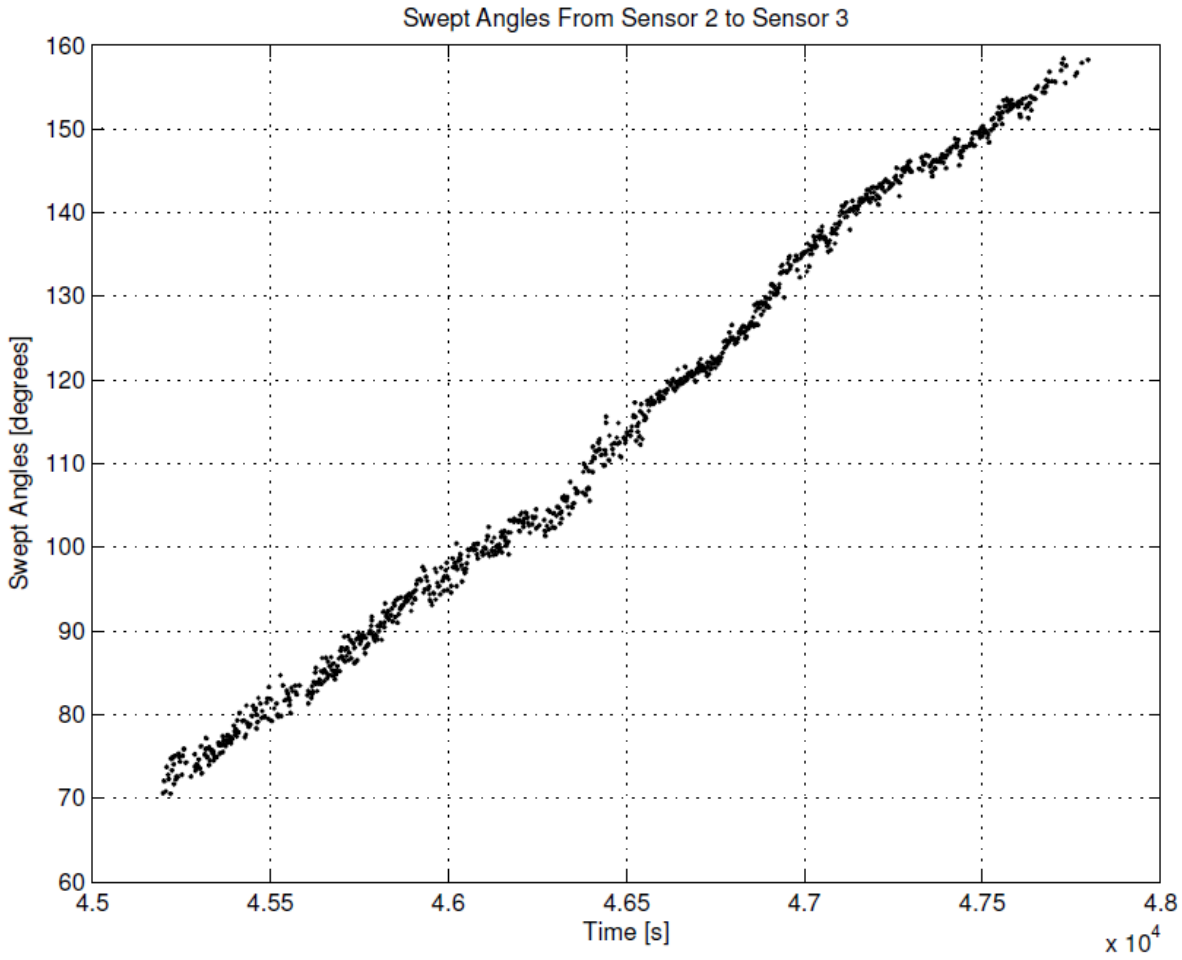

Figure 21. Calculated sweep angles between sensor 2 and sensor 3 over 50 minutes (45 000 - 48 000 seconds counting from 00:00 am on recording day).

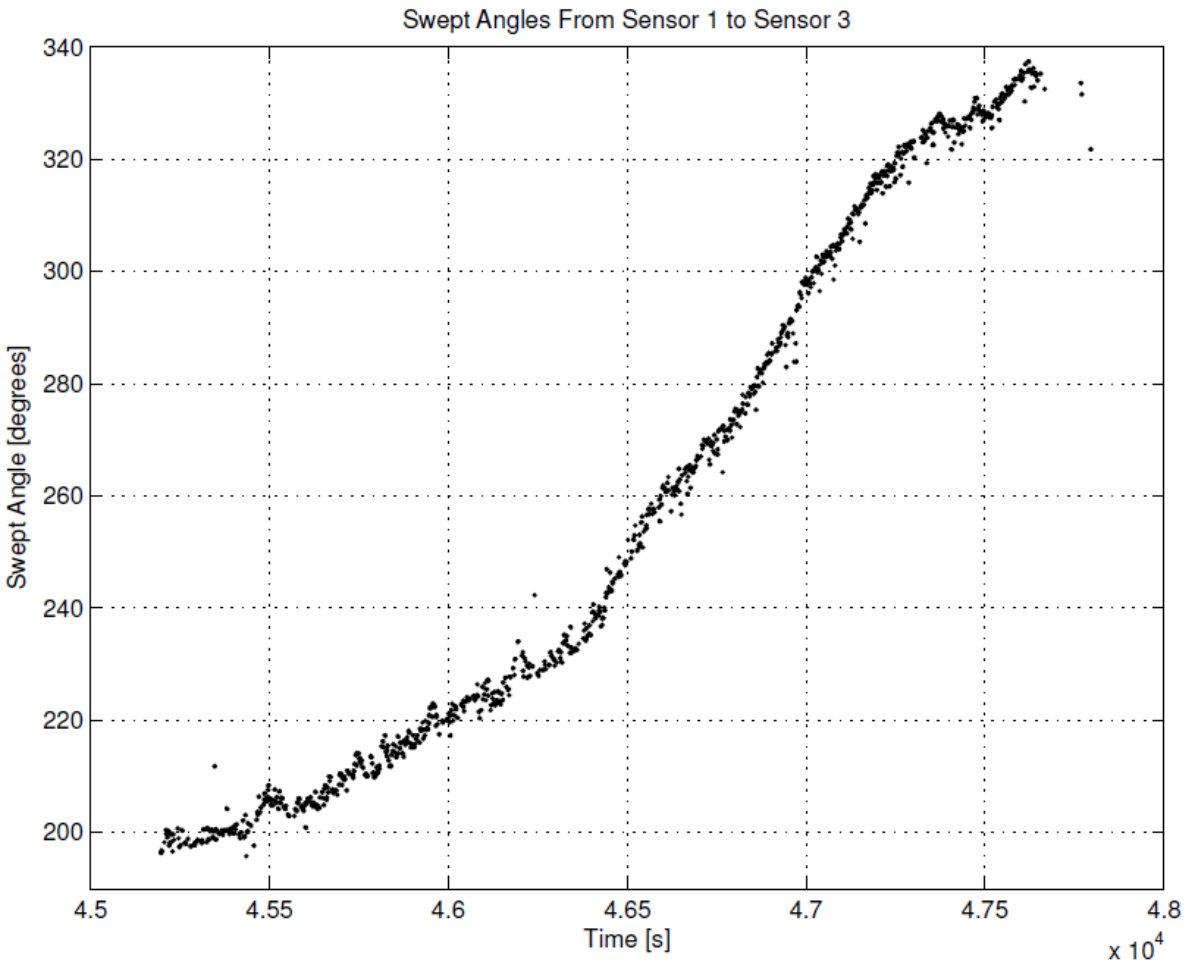

Figure 22. Calculated sweep angles between sensor 1 and sensor 3 over 50 minutes (45 000 - 48 000 seconds counting from 00:00 am on recording day).

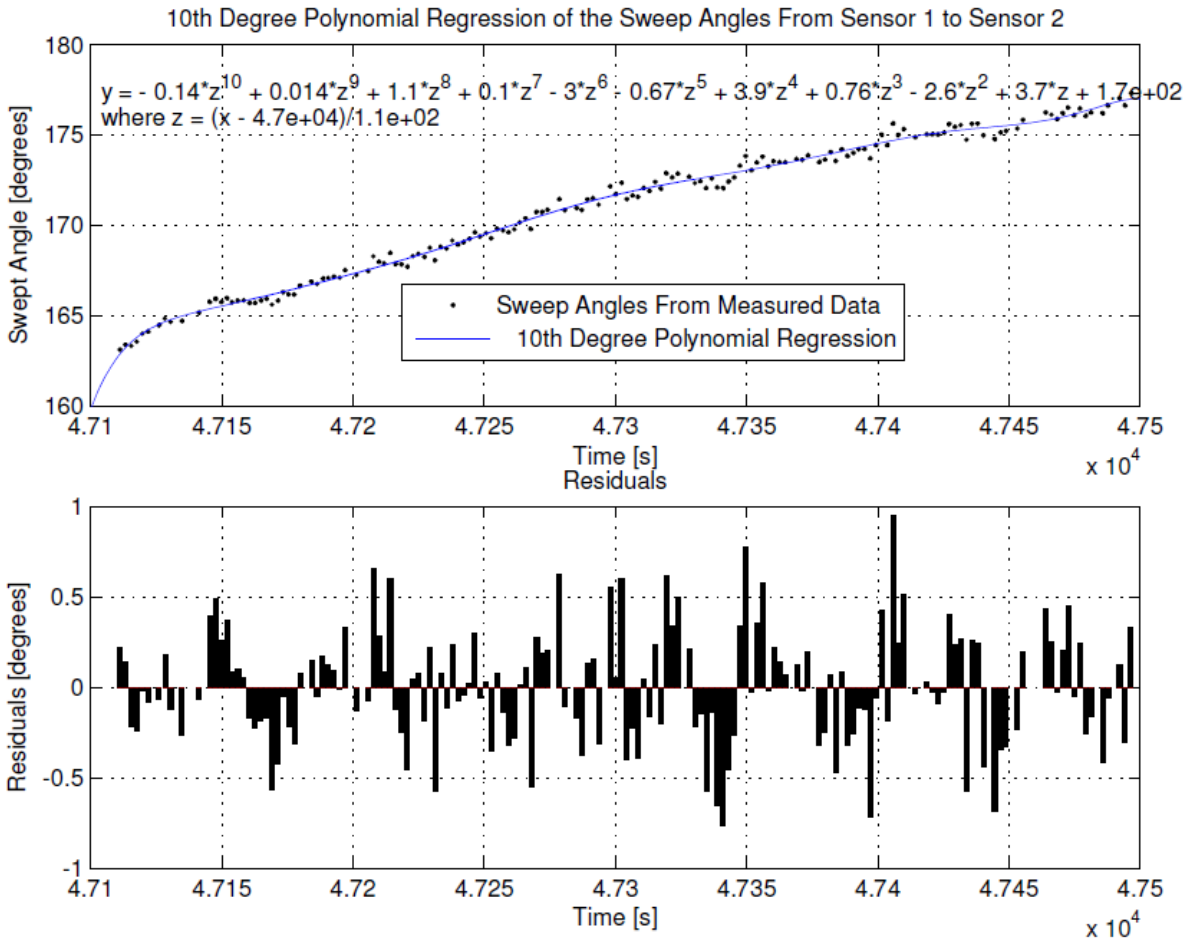

Figure 23. The upper diagram show the regression of the calculated sweep angles between sensor 1 and sensor 2, within a limited time range. The lower diagram show the residuals of the calculated sweep angles compared with the regression function. Std of resiudals = 0.3154°

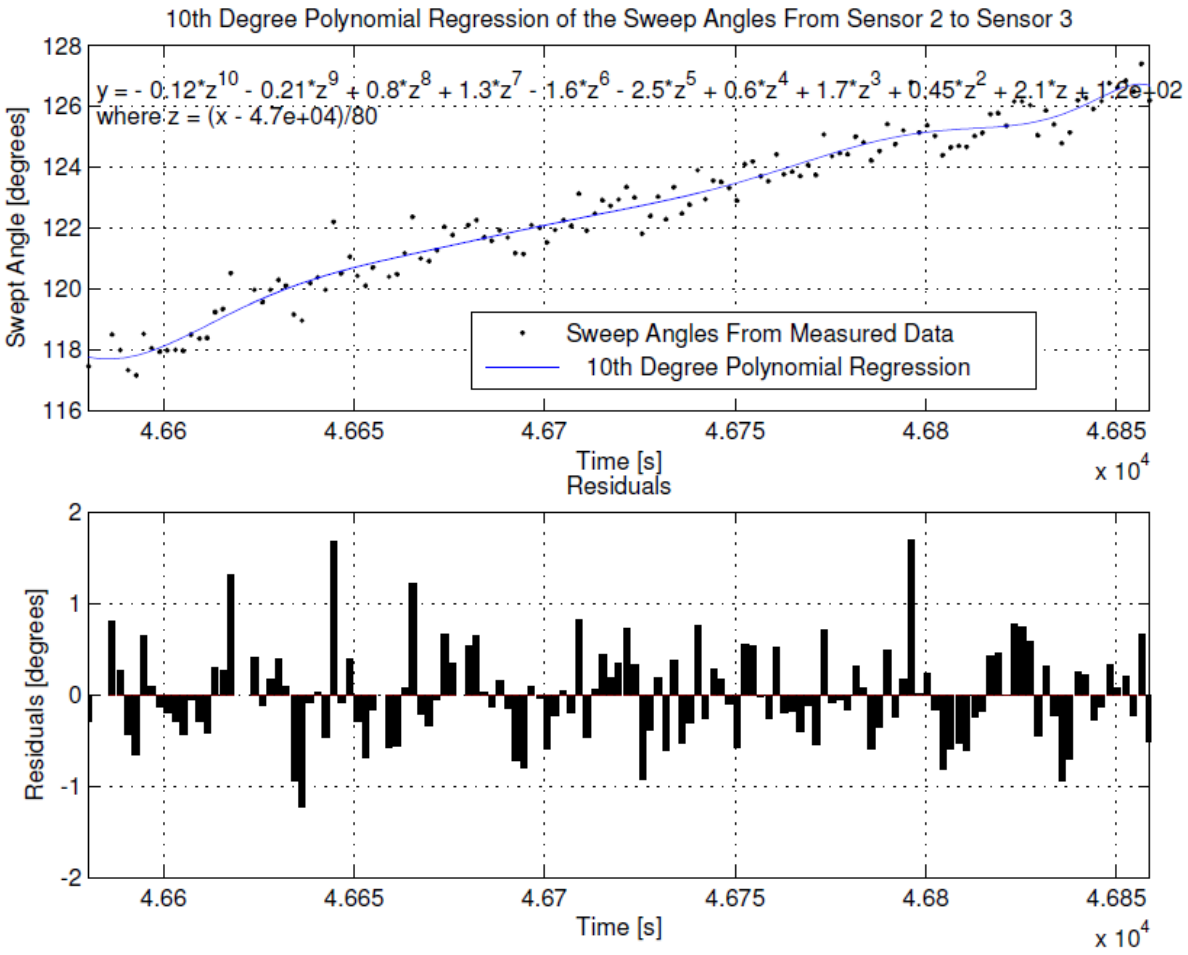

Figure 24. The upper diagram show the regression of the calculated sweep angles between sensor 2 and sensor 3, within a limited time range. The lower diagram show the residuals of the calculated sweep angles compared with the regression function. Std of resiudals = 0.5111°

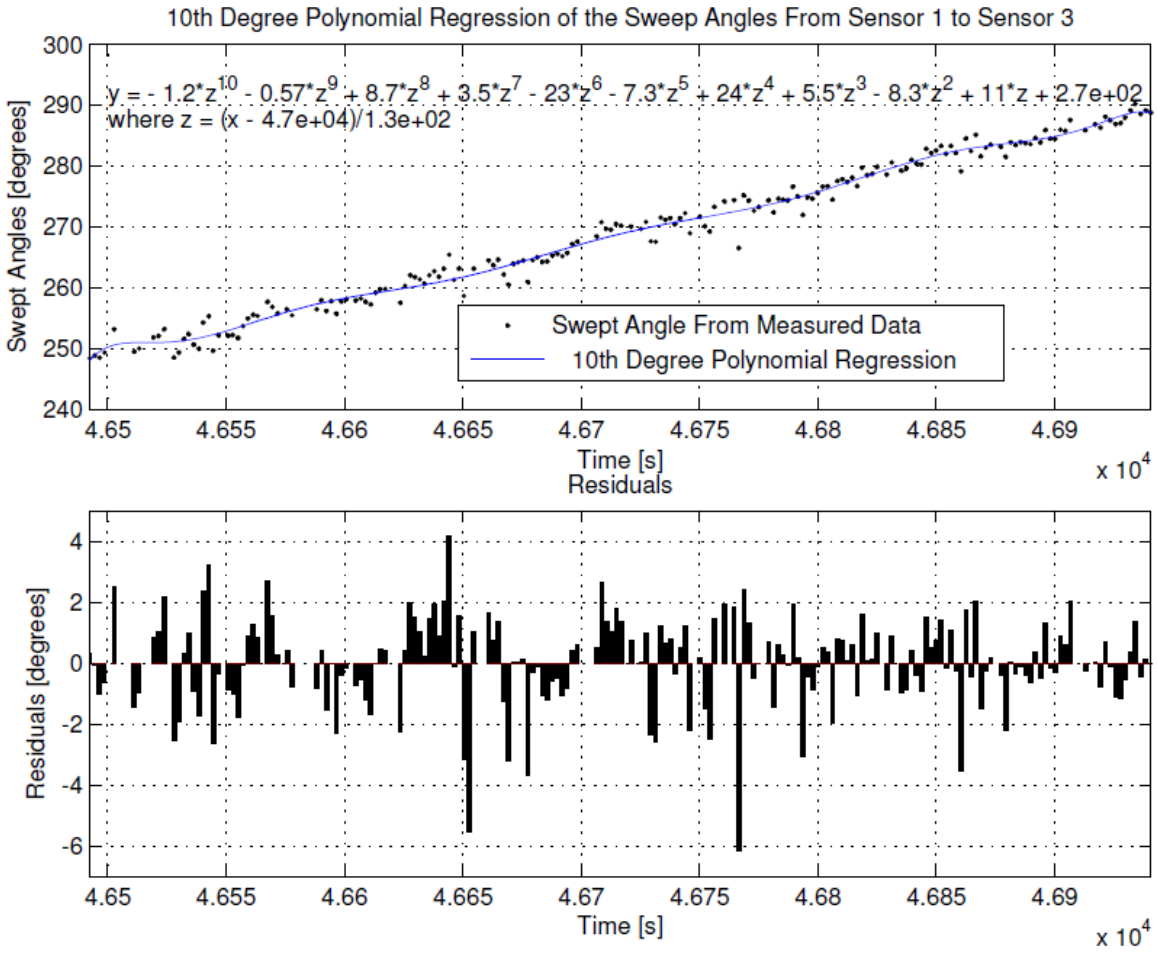

Figure 25. The upper diagram show the regression of the calculated sweep angles between sensor 1 and sensor 3, within a limited time range. The lower diagram show the residuals of the calculated sweep angles compared with the regression function. Std of resiudals = 1.4742°

Table 5

Std of Residuals

Sensor Set	std of Residuals
Sensor 2 → Sensor 1	0.3154°
Sensor 2 → Sensor 3	0.5111°
Sensor 1 → Sensor 3	1.4742°

Sensor Distribution Scale, Range and Geolocation Accuracy

Figure 26 illustrates the three performance indicators DSS was tested on. Table 6 show the sensor locations for the Finnmark Coast case study.

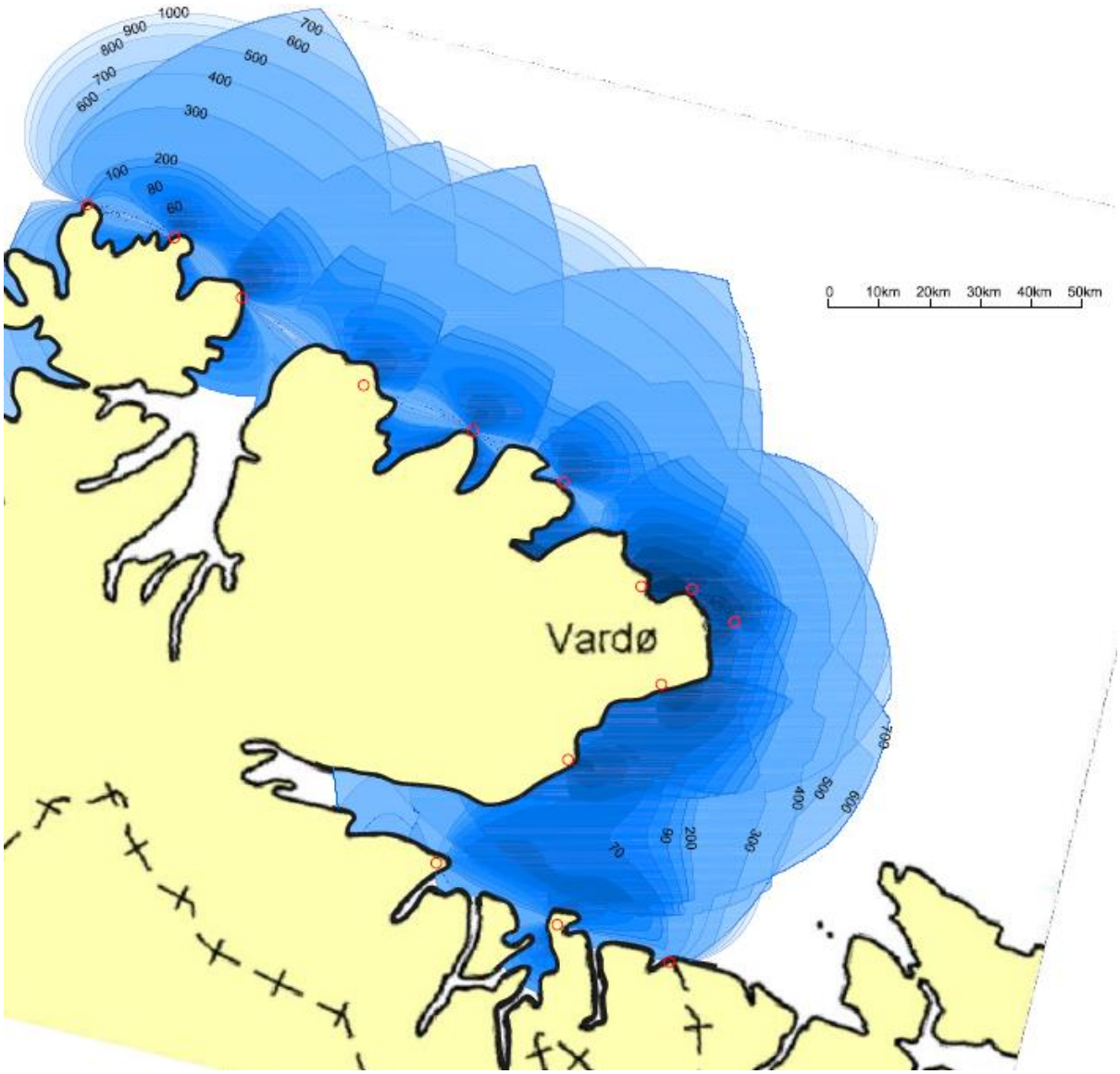

Figure 26. Estimate on DSS geolocation accuracy. The gradients of blue illustrates the accuracy and range of DSS outside the eastern Finnmark Coast in Norway by the Russian border. The accuracy is based on CEP with 0.45° standard deviation in the measured sweep angles. The contour labels mark the CEP value in meters. Red circles mark sensor positions.

Table 6

Sensor Locations

	Coordinates	Height
Sensor 1	[69.781, 30.790]	212 m.a.s.l
Sensor 2	[69.848, 30.210]	408 m.a.s.l
Sensor 3	[69.958, 29.583]	294 m.a.s.l
Sensor 4	[70.142, 30.271]	133 m.a.s.l
Sensor 5	[70.276, 30.763]	128 m.a.s.l
Sensor 6	[70.387, 31.152]	65 m.a.s.l
Sensor 7	[70.445, 30.925]	101 m.a.s.l
Sensor 8	[70.450, 30.654]	260 m.a.s.l
Sensor 9	[70.636, 30.237]	260 m.a.s.l
Sensor 10	[70.727, 29.748]	214 m.a.s.l
Sensor 11	[70.808, 29.159]	381 m.a.s.l
Sensor 12	[70.964, 28.494]	268 m.a.s.l
Sensor 13	[71.071, 28.125]	200 m.a.s.l
Sensor 14	[71.130, 27.645]	237 m.a.s.l

Discussion

Scan-Period Stability

It is clear from Figure 15, 16 and 17 that there is a significant instability in the scan-period. The standard deviation from the 50 minute analysis in Table 3 corresponds to almost 5° at mean scan rate. In light of this, assuming constant scan-rate for calculating sweep angles would result in poor geolocation accuracy.

The scan-period fluctuations are close to identical as illustrated in Figure 18 and quantified in Table 4. Notice the obvious false values in Figure 18 separated from the concentration of values, particularly originating from Sensor 1 and Sensor 2, whose absence

would give even higher correlation factors. The high correlation strongly suggests that the scan-period instability is caused by the nature of the navigation radar and not a cause of the sensing ability of the sensors or further signal processing. Several radars were detected in the same time and space. All radars investigated were found to have similar behaviour.

Analysing the scan-period over 5 minutes rather than 50 minutes (Figure 19, Table 2 and 4) gives a better stability. The standard deviation from the 5 minute analysis corresponds to about 0.8°. The correlation coefficients (Table 4) indicate that there is still a positive relationship between the three scan-period sets. It is therefore reasonable to believe that the standard deviation between real scan-period and measured scan-period is less than 0.21 %.

DSS will need a floating scan-period for the algorithm calculating ship positions to achieve appropriate geolocation accuracy. Based on the data in Figure 19 and associated text, I propose the scan-period to be calculated based on the 5 most recent recordings (about 11 seconds in the case of the subject radar).

Sweep Angles

It is promising that it was possible to retrieve three sets of sweep angles almost without interruption over 50 minutes. It should be noted that the subject radar was selected because of its promising stability on all three sensors. The time range of which the radar data was examined was also not randomly picked. In a perfect world, plotting the PRI over time should reveal a pattern of horizontal PRI-series were each series represented one radar. It is clear from Figure 9 that this was not the case in this experiment. The PRI-series are interrupted by long intervals without data, and it appears to be a lot of noise particularly on the sensor 2. The intervals without data are probably because of geometric limitations where the sensor line of sight is somehow broken. A full scale DSS would be able to rely on data from more sensors and, with strategic distribution, cover an area with less “shaded” areas.

There will always be some noise influencing the recorded radar signals. For this research the sensor data was investigated at the end of the recording. Live processing of the data demands different filtration strategy because trending is not possible. Dogancay (2007) describes an "Online Optimization of Receiver Trajectories", where the noise influence is considered. The majority noise originates from the sensor with highest proximity to the sea, both vertical and horizontal. It is possible that the noise measured on sensor 2 originates from radar signals reflected by the sea surface and surrounding buildings or disturbance from strong sidelobes.

Please be aware that the sweep angles presented in the Figure 20, AD and AE are not the actual geometric angles. This is because the sensors turned out not to be time synchronized before data recording was initiated. Efforts were made at FFI to synchronize the sensor recorded data after the measurement campaign. Absolute synchronization was not achieved during this work. However, according to FFI (2013), synchronization down to aproximately 1 second offset for one of the sensors succeeded (unable to determine which sensor). Two days before due date of the thesis FFI confirmed Sensor 2 to be exactly 1 second lead to Sensor 1 and Sensor 3. Unfortunately it was too late to account for this in the thesis. From Equation 3 it is clear that for the calculation of sweep angles, a one second offset produces completely wrong sweep angles for two of the sensor combinations, resulting in completely wrong ship locations. It will not affect the credibility of the scan-period in the same way because a typical change in scan-period from second to second is negligible compared to the scan-period (typically less than 2 ms or 0.94 ‰). It is assumed that the variance of the calculated scan-periods is not affected by the sensor time offset, because the offset is constant.

Sensor Distribution Scale, Range and Geolocation Accuracy

As a rule of thumb, it seems a distance between the sensors equal to half the sensor range give an appropriate sensor density when the sensor-to-sensor baseline is linear. Higher density is required around capes, and lower density can be accepted around bays. Exceptions may be made for a more favourable geological position i.e. higher elevation or better sea view. Following this rule and paying attention to the exceptions gives 14 sensors from the Russian border to North Cape (71°N), which is an effective coastline of approximately 285 km (adding distances between sensors). The sensor locations used in Figure 26 where found by studying the elevation profile of the terrain. It is not enough to choose the highest spot in a candidate area for a sensor location. It also need free sight down to the coastline, which means that the elevation profile down to the coast also must be considered. Otherwise there will be a shaded area from where the sensor cannot receive radar signals. Figure 27 illustrates this.

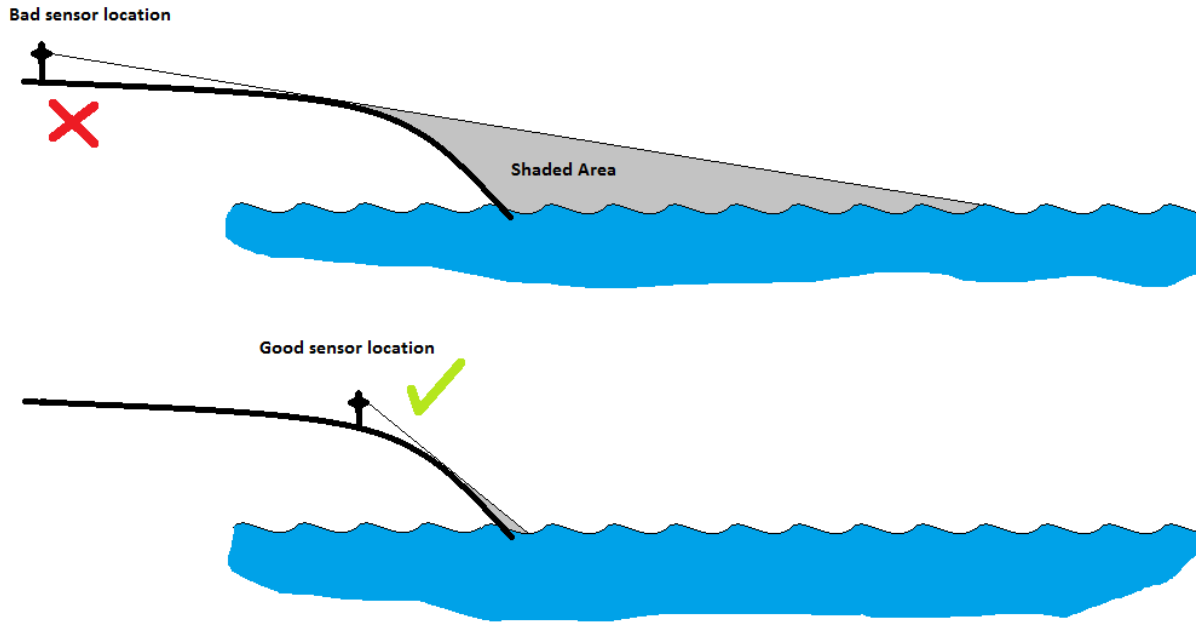

Figure 27. Shaded area caused by bad sensor location.

Although shaded areas were attempted avoided when choosing sensor location for Finnmark, some shaded areas must be expected in Figure 26 as the method of picking location based on studying of an elevation chart has its limitations. Considerations for avoiding shaded areas may affect the sensor density. The power availability was not considered when choosing sensor locations in Table 6. Some of the locations are desolate. Considering power availability may also affect sensor density.

With the sensor positions in Table 6, DSS has an operating range of 40-50 km from sensor baseline. The range varies based on availability of good elevated sensor locations along the coastline and the curvature of the coast.

The geolocation accuracy reveals itself in Figure 26. Based on the figure, DSS has a position accuracy of roughly 1000 meters CEP (50 % chance of calculating the ship position within a radius of 1000 meters) at a range of 40-50 km. At about 20 km the CEP is 100 meters. Notice that the accuracy is bad near the baseline between the sensors. This is an unfortunate phenomenon that might require an extra sensor.

It is obvious that the accuracy curves in Figure 26 does not describe the accuracy of the measurement setup from measurement campaign in the Oslofjord, nor was that the intention. The intention was to estimate the potential of the DSS in terms of sensor density, tracking range and geolocation accuracy. The next step will be to track ship movements live, and to check actual ship movement against calculated movement, and to test the tracking range.

As previously mentioned, there has been some trouble with the time synchronization of the sensors. Ship locations where not calculated in the research because of this. This is a weakness because it unable comparison between calculated location and actual location, which would say something about the validity and quality of the estimated geolocation accuracy illustrated in Figure 26.

Conclusion.

A DSS application in Finnmark has the potential of performing geolocation of ships with an accuracy of roughly 1000 m CEP at ranges up to 40 - 50 km from the coast, and 100 m 20 km from the coast. The accuracy was based on a sweep angle std of 0.45 degrees, which were obtained from the three radar sensors in the Oslofjord.

In the same DSS application, 14 were used to cover a 285 km effective coastline, from the North Cape to the Russian boarder. Average distance between sensors was 21.9 km.

References

Barton, D. K. (1988). *Modern Radar System Analysis.* Norwood, MA: Artech House inc.

Chang, L., & Xiaofei, S. (2009). Study of Data Fusion of AIS and Radar. *2009 International Conference of Soft computing and Pattern Recognition*, (pp. 674 - 677).

Dogancay, K. (2007). Online Optimization of Receiver Trajectories for Scan-Based Emitter Localization. *43*(3), pp. 1117 - 1125.

FFI. (2011). Kjeller, Norway. (Unpublished diagram.)

FFI. (2013). Kjeller, Norway. (Confirmation by E-mail.)

Grønvold, L., & Oftebro, S. (2006). *LabView program for lokalisering av båter med tidsinformasjon fra to sensorer [LabView program for localization of ships with time information from two sensors](Unpublished bachelor's thesis).* Oslo: Oslo University College.

Hmam, H. (2007, January). Scan Based Emitter Passive Localization. *IEEE Transactions on Aerospace and Electronic Systems, 43*(1), pp. 36-54.

Hope, B. (2013). *Prosjektbeskrivelse MAROFF141013 [Project Description MAROFF141013]*. Lommedalen: Sensorteknikk.

Høye, G. (2010). *Analyses of the geolocation accuracy that can be obtained from shipborne sensors by use of time difference of arrival (TDOA), scanphase, and angle of arrival (AOA) measurements.* Kjeller: FFI.

Kim, W. C., Song, T. L., & Musicki, D. (2012). Mobile Emitter Geolocation and Tracking Using Correlated Time Difference of Arrival Measurements. *2012 15th International Conferenceon Information Fusion*, (pp. 700-706).

Løvli, E. O. (2005). *Geolocalization with two airborne ESM sensors.(Unpublished master's thesis)* NTNU.

Mikhalev, A., & Ormondroyd, R. (2007). Passive Emitter Geolocation using Agent-based Data Fusion of AOA, TDOA and FDOA Measurements. *2007 10th International Conference on Information Fusion*, (pp. 1-6).

Smestad, T. (2013). *Geolokaliseringsnøyaktighet av Skip med kun Skannfase* [Geolocalization of Ships with Scan-Based Emitter Technique Only]. (Matlab code) Kjeller, Norway.

Teknisk Ukeblad. (2012). Skal Sikre Kysten med Sensorer [Will Secure the Coast with Sensors]. *TU, 12*(05).

Teknisk Ukeblad. (2013). Utvikler radarsensorer for skipsovervåkning [Developing radarsensors for ship surveillance]. *TU, 13*(05), 12-13.

Attachment A

```
%Matlab code

%Author:    Robert Engebråten
%Date:      25.10.2013
%Purpose:   Importing Lobe Description Words (LDW) for further analysis
%               - Identify radars
%               - Analyse radar scan-period stability
%               - Calculate the radar beam's swept angles between sensors
%Note:      Code is useless without the lobe files: Line1.lobes,
%           Line2.lobes and DIPPII.lobes


fid=fopen('Line1.lobes','r','l');
LokalID=int32([]);
LokalID=reshape(LokalID,0,62943 );
ToA = double([]);
ToA = reshape(ToA,0,62943 );
A = double([]);
A = reshape(A,0,62943 );
A(62943)=0;
RotTime=double([]);
RotTime=reshape(RotTime,0,62943 );
PRI=double([]);
PRI=reshape(PRI,0,62943 );
PeakAmp=double([]);
PeakAmp=reshape(PeakAmp,0,62943 );

for i=1:62943                % limited by DIPPII.lobes
    data=fread(fid,1,'int32');
    LokalID(i)=data;               % creating vector with Lokal ID
    %position=ftell(fid)
    data=fread(fid,1,'double');
    %data=data+1362009600;
    ToA(i)=data;                   % creating vector with Time of arrival
    data=fread(fid,1,'double');
    RotTime(i)=data;                 % creating vector with Rotation time
    data=fread(fid,1,'double');
    PRI(i)=data;                   % creating vector with PRI
    if (data > 0.00082) && (data < 0.000835)
        A(i)=ToA(i);
    end
    data=fread(fid,1,'uint32');
    PeakAmp(i)=data;                  % creating vector with pulse
amplitudes
end

fid=fopen('Line2.lobes','r','l');
LokalID_2=int32([]);
LokalID_2=reshape(LokalID_2,0,34059 );
ToA_2 = double([]);
ToA_2 = reshape(ToA_2,0,34059 );
B = double([]);
B = reshape(B,0,34059 );
B(34059)=0;
RotTime_2=double([]);
RotTime_2=reshape(RotTime_2,0,34059 );
PRI_2=double([]);
PRI_2=reshape(PRI_2,0,34059 );
```

```
PeakAmp_2=double([]);
PeakAmp_2=reshape(PeakAmp_2,0,34059 );

for i=1:34059                % limited by DIPPII.lobes
    data=fread(fid,1,'int32');
    LokalID_2(i)=data;               % creating vector with Lokal ID
    %position=ftell(fid)
    data=fread(fid,1,'double');
    data=data+1;
    ToA_2(i)=data;                   % creating vector with Time of arrival
    %position2=ftell(fid)
    data=fread(fid,1,'double');
    RotTime_2(i)=data;                 % creating vector with Rotation time
    data=fread(fid,1,'double');
    PRI_2(i)=data;                   % creating vector with PRI
    if (data > 0.00082) && (data < 0.000835)
        B(i)=ToA_2(i);
    end
    data=fread(fid,1,'uint32');
    PeakAmp_2(i)=data;                  % creating vector with pulse
amplitudes
end

fid=fopen('DIPPII.lobes','r','l');
LokalID_3=int32([]);
LokalID_3=reshape(LokalID_3,0,12904 );
ToA_3 = double([]);
ToA_3 = reshape(ToA_3,0,12904 );
C = double([]);
C = reshape(C,0,12904 );
C(12904)=0;
RotTime_3=double([]);
RotTime_3=reshape(RotTime_3,0,12904 );
PRI_3=double([]);
PRI_3=reshape(PRI_3,0,12904 );
PeakAmp_3=double([]);
PeakAmp_3=reshape(PeakAmp_3,0,12904 );

for i=1:12904                % limited by DIPPII.lobes
    data=fread(fid,1,'int32');
    LokalID_3(i)=data;               % creating vector with Lokal ID
    %position=ftell(fid)
    data=fread(fid,1,'double');
    ToA_3(i)=data;                   % creating vector with Time of arrival
    %position2=ftell(fid)
    data=fread(fid,1,'double');
    RotTime_3(i)=data;                 % creating vector with Rotation time
    data=fread(fid,1,'double');
    PRI_3(i)=data;                   % creating vector with PRI
    if (data > 0.00082) && (data < 0.000835)
        C(i)=ToA_3(i);
    end
    data=fread(fid,1,'uint32');
    PeakAmp_3(i)=data;                  % creating vector with pulse
amplitudes
end


dogn=60*60*24;
ToA=mod(ToA,dogn);
```

```
ToA_2=mod(ToA_2,dogn);
ToA_3=mod(ToA_3,dogn);

A=mod(A,dogn);
B=mod(B,dogn);
C=mod(C,dogn);


Result1=load('Res1.txt','-ascii','double','format long'); %load sample of
ToAs whit same PRI
Result2=load('Res2.txt','-ascii','double','format long'); %load sample of
ToAs whit same PRI
Result3=load('Res3.txt','-ascii','double','format long'); %load sample of
ToAs whit same PRI
Rotation1=load('Rot1.txt','-ascii','double','format long'); %load matching
rotation time
Rotation2=load('Rot2.txt','-ascii','double','format long'); %load matching
rotation time
Rotation3=load('Rot3.txt','-ascii','double','format long'); %load matching
rotation time
Rotfunction=load('Rotfunction.txt','-ascii','double','format long'); %load
best function of rotation time

Result1=sort(Result1);
Result2=sort(Result2);
Result3=sort(Result3);

TimeDiff_12=double([]);
TimeDiff_12=reshape(TimeDiff_12,0,930);
ScanPhaseAngle_12=double([]);
ScanPhaseAngle_12=reshape(ScanPhaseAngle_12,0,930);

i=1;
j=1;
t=1;
k=1;
checkB=0;
checkA=0;


while k==1
    difference=(Result2(j)-Result1(i));

    if (difference>1.2)||(difference<0.0)
        while (difference > 1.2)||(difference < 0.0)
            difference=(Result2(j)-Result1(i));
            CheckA=(difference);
            j=j+1;
            difference=(Result2(j)-Result1(i));
            CheckB=(difference);

            if (CheckB>CheckA)&&(difference>1.2)
                j=j-1;
                difference=1.0;
            end
            if (CheckB<CheckA)&&(difference<0)
                j=j-1;
                difference=1.0;
            end
```

```
        end
        while (difference > 1.2)||(difference < 0.2)
            difference=(Result2(j)-Result1(i));
            CheckA=(difference);
            i=i+1;
            difference=(Result2(j)-Result1(i));
            CheckB=(difference);
            if (CheckB>CheckA)&&(difference>1.2)
                i=i-1;
                print('something is wrong')
                difference=1.0;
            end
            if (CheckB<CheckA)&&(difference<0)
                j=j-1;
                difference=1.0;
            end
        end
    end

    difference=abs(Result2(j)-Result1(i));
    if t>16
        difference=Result2(j)-Result1(i);
        start=t-16;
        slutt=t-1;
        sample=TimeDiff_12(start:slutt);
        Median=median(sample);
        hoymedian=1.05*abs(Median);
        lavmedian=abs(Median)/1.05;
        if (difference>Median-0.1)&&(difference<Median+0.1)

            TimeDiff_12(t)=Result2(j)-Result1(i);
            Time_12(t)=Result1(i)+(TimeDiff_12(t))/2; %Problemer med
vektorlengde

            t=t+1;

        end

    end
    %må ligge i nærheten av de 15 foregående

    if t<=16
        if difference<=1.2

            TimeDiff_12(t)=Result2(j)-Result1(i);
            Time_12(t)=Result1(i)+(TimeDiff_12(t))/2; %Problemer med
vektorlengde
            t=t+1;
        end
    end
    i=i+1;

    if i>=length(Result1)
        k=0;
    end
    if j>=length(Result2)-1
        k=0;
    end
```

```
end

for i=1:length(TimeDiff_12);
    factor=length(Rotfunction)/length(TimeDiff_12);
    modulus=mod(factor,1);
    factor=factor-modulus;
    Rotationtime=Rotfunction(1+(i-1)*factor);
    ScanPhaseAngle_12(i)=360*TimeDiff_12(i)/Rotationtime;
end

figure
scatter(Time_12,ScanPhaseAngle_12,'k.','linewidth',1); hold on;
%     plotResiduals(ans);

TimeDiff_23=double([]);
TimeDiff_23=reshape(TimeDiff_23,0,930);
ScanPhaseAngle_23=double([]);
ScanPhaseAngle_23=reshape(ScanPhaseAngle_23,0,930);

i=1;
j=1;
t=1;
k=1;
checkB=0;
checkA=0;

while k==1
    difference=(Result3(j)-Result2(i));

    if (difference>1.2)||(difference<0.0)
        while (difference > 1.2)||(difference < 0.0)
            difference=(Result3(j)-Result2(i));
            CheckA=(difference);
            j=j+1;
            difference=(Result3(j)-Result2(i));
            CheckB=(difference);

            if (CheckB>CheckA)&&(difference>1.2)
                j=j-1;
                difference=1.0;
            end
            if (CheckB<CheckA)&&(difference<0)
                j=j-1;
                difference=1.0;
            end

        end
        while (difference > 1.2)||(difference < 0.2)
            difference=(Result3(j)-Result2(i));
            CheckA=(difference);
            i=i+1;
            difference=(Result3(j)-Result2(i));
            CheckB=(difference);
            if (CheckB>CheckA)&&(difference>1.2)
                i=i-1;
                print('something is wrong')
                difference=1.0;
            end
            if (CheckB<CheckA)&&(difference<0)
                j=j-1;
```

```
                difference=1.0;
            end
        end
    end

    difference=abs(Result3(j)-Result2(i));
    if t>16
        difference=Result3(j)-Result2(i);
        start=t-16;
        slutt=t-1;
        sample=TimeDiff_23(start:slutt);
        Median=median(sample);
        hoymedian=1.05*abs(Median);
        lavmedian=abs(Median)/1.05;
        if (difference>Median-0.01)&&(difference<Median+0.1)

            TimeDiff_23(t)=Result3(j)-Result2(i);
            Time_23(t)=Result2(i)+(TimeDiff_23(t))/2; %Problemer med
vektorlengde

            t=t+1;

        end

    end
    %må ligge i nærheten av de 15 foregående

    if t<=16
        if difference<=1.2

            TimeDiff_23(t)=Result3(j)-Result2(i);
            Time_23(t)=Result2(i)+(TimeDiff_23(t))/2; %Problemer med
vektorlengde
            t=t+1;
        end
    end
    i=i+1;

    if i>=length(Result2)
        k=0;
    end
    if j>=length(Result3)-1
        k=0;
    end
end

for i=1:length(TimeDiff_23);
    factor=length(Rotfunction)/length(TimeDiff_23);
    modulus=mod(factor,1)
    factor=factor-modulus;
    Rotationtime=Rotfunction(1+(i-1)*factor)
    ScanPhaseAngle_23(i)=360*TimeDiff_23(i)/Rotationtime;
end

figure
scatter(Time_23,ScanPhaseAngle_23,'r.','linewidth',1); hold on;

TimeDiff_13=double([]);
TimeDiff_13=reshape(TimeDiff_13,0,930);
```

```
ScanPhaseAngle_13=double([]);
ScanPhaseAngle_13=reshape(ScanPhaseAngle_13,0,930);

i=1;
j=1;
t=1;
k=1;
checkB=0;
checkA=0;


while k==1
    difference=(Result3(j)-Result1(i));

    if (difference>2.0)||(difference<0.0)
        while (difference > 2.0)||(difference < 0.0)
            difference=(Result3(j)-Result1(i));
            CheckA=(difference);
            j=j+1;
            difference=(Result3(j)-Result1(i));
            CheckB=(difference);

            if (CheckB>CheckA)&&(difference>2.0)
                j=j-1;
                difference=1.0;
            end
            if (CheckB<CheckA)&&(difference<0)
                j=j-1;
                difference=1.0;
            end

        end
        while (difference > 2.0)||(difference < 0.2)
            difference=(Result3(j)-Result1(i));
            CheckA=(difference);
            i=i+1;
            difference=(Result3(j)-Result1(i));
            CheckB=(difference);
            if (CheckB>CheckA)&&(difference>2.0)
                i=i-1;
                print('something is wrong')
                difference=1.0;
            end
            if (CheckB<CheckA)&&(difference<0)
                j=j-1;
                difference=1.0;
            end
        end
    end

    difference=abs(Result3(j)-Result1(i));
    if t>30
        difference=Result3(j)-Result1(i);
        start=t-30;
        slutt=t-1;
        sample=TimeDiff_13(start:slutt);
        Median=median(sample);
        hoymedian=1.05*abs(Median);
        lavmedian=abs(Median)/1.05;
        if (difference>Median-0.1)&&(difference<Median+0.1)
```

```
            TimeDiff_13(t)=Result3(j)-Result1(i);
            Time_13(t)=Result1(i)+(TimeDiff_13(t))/2; %Problemer med
vektorlengde

            t=t+1;

        end

    end
    %må ligge i nærheten av de 15 foregående

    if t<=30
        if difference<=1.2

            TimeDiff_13(t)=Result3(j)-Result1(i);
            Time_13(t)=Result1(i)+(TimeDiff_13(t))/2; %Problemer med
vektorlengde
            t=t+1;
        end
    end
    i=i+1;

    if i>=length(Result1)
        k=0;
    end
    if j>=length(Result3)-1
        k=0;
    end
end

for i=1:length(TimeDiff_13);
    factor=length(Rotfunction)/length(TimeDiff_13);
    modulus=mod(factor,1)
    factor=factor-modulus;
    Rotationtime=Rotfunction(1+(i-1)*factor)
    ScanPhaseAngle_13(i)=360*TimeDiff_13(i)/Rotationtime;
end

figure
scatter(Time_13,ScanPhaseAngle_13,'b.','linewidth',1); hold on;
```

Attachment B

```
% GEOLOKALISERINGSNØYAKTIGHET AV SKIP MED KUN SKANNFASE
% Tore Smestad, FFI 11/1-2013, mod mhp R (sensorrekkevidde) 28/5-13
%               mod for posisjonsberegning 30/5-13
%               mod for kov>0 og tall i konturplott 6/6-13
%
% Dette programmet er laget som et utgangspunkt for arbeid som student
% Robert Engebråten skal gjøre i sin masteroppgave ved Høgskolen i Vestfold
% våren 2013 veiledet av FFI i samarbeid med Bjørn Hope i Sensorteknikk.

% Hensikten med programmet er å kartlegge posisjonsnøyaktighet av skip med
% enkle sensorer som kan detektere skipenes navigasonsradar og avlede
% forskjellige type målinger. Dette programmet bruker kun på skannfase.
% Programmet kan analysere posisjonsnøyaktighet på flere måter:
% 1) Tegne "stedlinjer, usikkerhetsbånd og -område" for skannfasemålinger
% 2) Tegne usikkerhetsellipser for geolokalisering med skannfasemålinger
% 3) Tegne kontur-kart for CEP-verdier ut fra beregnede feilellipser
%
% "Stedlinjer" er posisjoner som gir samme skannfase-verdi, "usikkerhets-
% bånd" er området mellom stedlinjer med +/- måleusikkerhet, "usikkerhets-
% område" er skjæringsområdet mellom usikkehetsbånd. Feilellipser faller
% ofte sammen med usikkerhetsområder. Feilellipser er beregnet med "CRLB"
% (Cramer Rao Lower Bound) som er en nedre grense for feil i en hvilken som
% helst (forventningsrett) estimator. (Gode estimatorer oppnår tilnærmet
% denne nøyaktigheten).
%
% CEP (Circular Error Probable) er radien i en sirkel som inneholder 50%
% av alle tilfellene (tilsvarer omtrent største halvakse i en 1 sigma feil-
% ellipse som er noe flattrykt - ligger mellom 0.68 og 1.18 av største
% halvakse avhengig av hvor flattrykt ellipsen er).
%
% Posisjoner blir beregnet for skips (fasit-)posisjoner
% Det blir trukket normalfordelte feil ihht spesifisert usikkerhet
% Beregnede posisjoner kan da sammenlignes med feilellipsene

global SensPar SensPos SFVm;

%SensPos(i): [ 1:x(m) 2:y(m) 3: R(m)];% Sensorposisjonene
% SensPos= [[0      0   50e3]' ...
%     [1e3   -7e3  50e3]'...
%     [2e3   17e3  50e3]']'; %Sensorposisjoner

 SensPos=[[         0e5                    0e5   0.572740000000000e5]' ...
  [-0.222131238551124e5   0.074364083397545e5   0.773820000000000e5]' ...
  [-0.461252230738928e5   0.196454056606533e5   0.664960000000000e5]' ...
  [-0.200711391001188e5   0.400676428143816e5   0.464770000000000e5]' ...
  [-0.015819407886682e5   0.549402553124458e5   0.457000000000000e5]' ...
  [ 0.129496297830790e5   0.672600438259902e5   0.341010000000000e5]' ...
  [ 0.045047656125224e5   0.736973840402823e5   0.411890000000000e5]' ...
  [-0.055613945544537e5   0.742523259612699e5   0.628520000000000e5]' ...
  [-0.209777331965718e5   0.948961615758573e5   0.628520000000000e5]' ...
  [-0.389321573411568e5   1.049960329434850e5   0.575180000000000e5]' ...
  [-0.604655359237196e5   1.139859981592670e5   0.749580000000000e5]' ...
  [-0.846329841625328e5   1.312999537025641e5   0.637300000000000e5]' ...
  [-0.979543764630205e5   1.431754836863048e5   0.557830000000000e5]' ...
  [-1.152099865326177e5   1.497236476501779e5
0.602500000000000e5]']';%Sensorposisjoner

[Nsens d0]= size(SensPos);
```

```
Npar= Nsens*(Nsens-1)/2; % Antall par av sensorene

SensPar= zeros(Npar,2); % Alle par-sammensetninger av sensorer
iPar= 0;
for i=1:Nsens-1,
    for j=i+1:Nsens,
        iPar= iPar+1;
        SensPar(iPar,1)= i;
        SensPar(iPar,2)= j;
    end;
end; % i=1:Nsens-1,

% Skipsposisjoner for beregning/visning: Pos(i)= [Øst(m),Nord(m)]:
Skip= [[4e3 -7.5e3]' [12.25e3 5.25e3]' [10e3 11e3]' ]';

Fig= [1,0,0,1,0]; % Info som blir tegnet (ut fra om indeks i er lik 1):
% 1: Plott "Kart" med sensorposisjoner og skipsposisjoner + evnt annet
% 2: Plott usikkerhetsområder
% 3: Plott feilellipser
% 4: Plott CEP-konturer
% 5: Plotte posisjonsberegninger

%OmrHj= [[0 4e4]' [4.1e4 9e4]' ]'; % Områdets 2 hjørner (beregningene)
OmrHj= [[-15e4 -2e4]' [5e4 20e4]' ]'; % Områdets 2 hjørner (beregningene)
Ngrid= 800; % Antall grid for beregningsområdet (likt for x og y)

stdSFV= 0.3154;    % 1 std Skannfasevinke på en sensor (grader)
Nposcalc= 100;   % Antall pos.beregninger (gitt målefeil) pr skipsposisjon
randn('state',5); % initier randomgenerator (kan også gjøres i løkka)

% Plott sensorene m/rekkevidde, radarene  -----------------
if Fig(1)==1
    figure
    plot(SensPos(:,1),SensPos(:,2),'ko','linewidth',1); hold on;
    plot(SensPos(:,1),SensPos(:,2),'kx','linewidth',1); hold on;
    for k=1:Nsens,
        Circ(SensPos(k,1),SensPos(k,2),SensPos(k,3),1,0);
    end;
    plot(Skip(:,1),Skip(:,2),'k*','linewidth',1); hold on;
    title('Lokalisering av skip med skannfasemålinger');
    xlabel('Øst (m)')
    ylabel('Nord (m)')
    axis equal
    grid on
end;


%-----------------------------------------------------------------
% Beregn målinger og CRLB for spesifiserte sensorer i gitt område

SFV=    zeros(Npar); % Skannfasemåling for hvert sensorpar (fasit)
SFVm=   zeros(Npar); % Skannfasemåling for hvert sensorpar (med feil)
skff=  zeros(Nsens); % Feil i hver sensors skannfase
Gr_SFV=    zeros(Npar,Ngrid,Ngrid); % Skannfasemålinger i grid
Gr_CEP =   zeros(Ngrid,Ngrid); % CEP (alle oper. sensorer og målinger)
Gx= zeros(1,Ngrid); % Grid-verdier i x
Gy= zeros(1,Ngrid); % Grid-verdier i y
X = zeros(1,Ngrid); % (for plotterutine)
Y = zeros(1,Ngrid); % (for plotterutine)
```

```
dx= (OmrHj(2,1)-OmrHj(1,1))/(Ngrid-1);
dy= (OmrHj(2,2)-OmrHj(1,2))/(Ngrid-1);

y= OmrHj(1,2)-dy; % start for y-retning
for iy=1:Ngrid, % trinne i Nord-retning

    if fix(iy/10)*10 == iy % utskrift hvert 10. trinn
        fprintf( 1, 'iy: %d av: %d \n',iy, Ngrid);
    end
    y=y+dy; Gy(iy)= y; Y(iy)=iy; % Grid-ref
    x= OmrHj(1,1)-dx; % start for x-retning
    for ix= 1:Ngrid, % trinne i Øst-retning
        x= x+dx;
        Gx(ix)= x;  X(ix)=ix; % Grid-ref

        for ip=1:Npar,
            Gr_SFV(ip,ix,iy)= mSFV(ip,SensPar,SensPos,x,y); %Skannfase-
verdier
        end;

        [P,D]= Px_SFParRr(SensPar,SensPos,stdSFV,x,y);
        [CEP,iunder]= CEPfraKov0(P);
        Gr_CEP(ix,iy)= CEP;
    end; % ix= 1:Nx, % trinne i Øst-retning

end; % iy=1:Ngrid, % trinne i Nord-retning

% Plott usikkerhetsområder for alle par av skannfasemålinger mot skip(I)
I=2; % skipet som det måles mot

if Fig(2)==1
    for ip=1:Npar,
        Gr_SFVi= zeros(Ngrid,Ngrid);
        for i=1:Ngrid, Gr_SFVi(i,:)= Gr_SFV(ip,i,:); end;
        sfv= mSFV(ip,SensPar,SensPos,Skip(I,1),Skip(I,2));
        contour(Gx,Gy,Gr_SFVi(X,Y)',[sfv+stdSFV sfv+stdSFV],'k:'); hold on;
        contour(Gx,Gy,Gr_SFVi(X,Y)',[sfv-stdSFV sfv-stdSFV],'k:'); hold on;
    end;
end;

if Fig(3)==1 % Plott feilellipse for utvalgt skip
    [P,D]= Px_SFPar(SensPar,SensPos,stdSFV,Skip(I,1),Skip(I,2));
    PlElli2D(P,[Skip(I,1),Skip(I,2)],1);
    PlElli2D(P,[Skip(I,1),Skip(I,2)],2);
end;

if Fig(4)==1 % Plott CEP-konturer (her 10m og 100m konturer)
    %pl2x10CountCRD(Gr_CEP,Gx,Gy,Ngrid,Ngrid,100);
    %     [C,H]= pl2x10CountCRDr(Gr_CEP,Gx,Gy,Ngrid,Ngrid,100);
    %     clabel(C,H);
    V= zeros(1,10); for i=1:10; V(i)=10*i; end; % Kontur-verdier
    [C,H] =contour(Gx,Gy,Gr_CEP(X,Y)',V,'k:'); hold on;

    clabel(C,H);
    V= zeros(1,10); for i=2:10; V(i)=100*i; end; % Kontur-verdier
    [C1,H1] =contour(Gx,Gy,Gr_CEP(X,Y)',V,'k:'); hold on;
    clabel(C1,H1);
end;
```

```
if Fig(5)==1 % Beregne og plotte posisjonsberegninger
    iSk= I; % Indeks for skip som det skal beregnes posisjon for
    for ip=1:Npar, % fasit skannfase for skipsposisjonen
        SFV(ip)= mSFV(ip,SensPar,SensPos,Skip(iSk,1),Skip(iSk,2));
    end;
    for iMC= 1:Nposcalc, % Posisjonsberegning pr trukket feil i skannfase

        %randn('state',3*iMC); % initier randomgenerator hver Monte Carlo
        for is=1:Nsens, skff(is)= stdSFV*randn; end; % trekking av feil
        for ip=1:Npar, % lag skannfasemålinger med feil
            SFVm(ip)= SFV(ip)+ skff(SensPar(ip,1))+ skff(SensPar(ip,2));
        end;

        % MATLAB-funksjon for minimering av kvadratavvik def ved "LFunk")
        [xc,dev]= fminsearch('LFunk',[Skip(iSk,1),Skip(iSk,2)]);
        posE= xc(1); posN= xc(2); % Beregnede posisjoner
        plot(posE,posN,'k+','linewidth',1); hold on;

    end;
end;
```

Function pl12x10CountCRDr

```
function [C,H]= pl2x10CountCRDr(CRD_Gr,Gx,Gy,Nxgrid,Nygrid,mdec)
% PLOT OF CONTOUR DIAGRAM FOR CHANGE IN RANGE DIFFERENCE 2x10 VAL
% Tore Smestad 4/1-08
%
% CRD_Gr: The "height landscape" to plot contours of
% Gx,Gy:  The grid points to use
% Nxgrid,Nygrid: The number of grid points in x and y
% mdec: Middle decade to use in plot, scales down to 0.1 and up to 10

X= zeros(1,Nxgrid);
Y= zeros(1,Nygrid);

Ndv= 10; % Nuber of steps up and down in decades
Dv= mdec; % Dv is the middle decade value
dv= mdec/Ndv;
%NDv=10;        % NB dv*Ndv =! Dv (når sc=1)
V= zeros(1,Ndv);

for i=1:Ndv,
   V(i)= dv*i; % Values of contours
end;

%figure
for ix=1:Nxgrid,
   X(ix)=ix;
end;
for iy=1:Nygrid,
   Y(iy)=iy;
end;

[C,H] =contour(Gx,Gy,CRD_Gr(X,Y)',V,'k:'); hold on;
for i=1:Ndv,
```

```
   contour(Gx,Gy,CRD_Gr(X,Y)',[i*Dv i*Dv],'k-'); hold on;
end;

%title('contour +/- values in 10 steps 0.01m and 10 steps in 0.1m ');
% xlabel('Position East (m)');
% ylabel('Position North (m)');

dummy=1;
return
```

Function PlElli2D

```
function PlElli2D(P,fly,asigm)
% PLOTTING AV SIGMA-ELLIPSE I 2D
% TSm 24/8-00 basert på tidligere
%

c1=180/pi;
%P
%fly
Pin= inv(P);    % Invertert kovariansmatrise (2x2)

xfly= fly(1); yfly= fly(2);

% Plott ellipse
dv = 0.5;       % step size for cv
cv = 1./tan([dv:dv:180-dv]/c1); % vinkel 0 og 180 settes spesielt

x1 = [ 0 ...
      sqrt( 1./(Pin(1,1) + 2*Pin(1,2).*cv + Pin(2,2)*cv.^2) )...
      0 ];
x2 = [ sqrt( 1/Pin(2,2) )*sign(x1(2))...
      cv.*x1(2:end-1) ...
      -sqrt( 1/Pin(2,2) )*sign(x1(2)) ];

%plot(asigm*x1+xfly,asigm*x2+yfly,'k','linewidth',1.5); hold on;
%plot(-asigm*x1+xfly,-asigm*x2+yfly,'k','linewidth',1.5); hold on;
plot(asigm*x1+xfly,asigm*x2+yfly,'k','linewidth',1.5); hold on;
plot(-asigm*x1+xfly,-asigm*x2+yfly,'k','linewidth',1.5); hold on;

return
```

Function Px_SFParRr

```
function [P,D]= Px_SFParRr(SensPar,SensPos,stdSFV,rx,ry)
% KOVARIANSMATRISE FOR SKANNFASE-GEOLOKALISERING
% TSm 8/1-2013, basert på tidligere program
%               mod mhp R (sensorrekkevidde) 29/5-13
%               mod "robust mot minst 3 sensorer"
%
% INN:
% SensPar: angivelse av sensorpar (ref til SensPos)
% SensPos: sensorenes posisjon
% stdSFV: måleusikkerhet for en skannfase (grader)
% rx,ry: posisjonen til radaren
```

```
% UT:
% P:    kovariansmatrise (2 x 2) - usikkerheten i radarens posisjon
% D:    målematrise (for inspeksjon i kommandovinduet)
%       rad: skannfasemålinge (sensorparene)
%       kol: tilstandsvariablene, x,y

% Robusthet mot for få sensorer er gjort ved at en bestandig måler
% posisjon nord og øst, men med dårlig nøyaktighet 100km (1 sigma)

[Nsens d0]= size(SensPos);
Npar= Nsens*(Nsens-1)/2; % Antall par av sensorene

Ndimx=2;        % antall tilstandsvariable (øst,nord for ukjent radar)

Pin= zeros(Ndimx);      % Invertert kovariansmatrise
%D=     zeros(Npar,Ndimx);  % Målematrise
D=  zeros(Npar+2,Ndimx);    % Målematrise (2 ekstra "målinger")
%Pwi=   zeros(Npar);    % Invertert målestøymatrise
Pwi=    zeros(Npar+2);  % Invertert målestøymatrise (2 ekstra "målinger")

dx=1; dy=1;     % størrelsen av pertubasjonene (m)

ilp= 0; % løpende nummer for sensorpar innenfor sensorenes rekkevidde
for ip=1:Npar, % antall sensorpar for skannfasemåling (ikke uavhengige?!)

    is1= SensPar(ip,1); is2= SensPar(ip,2); % sensorene som inngår i paret
    R1= sqrt((SensPos(is1,1)- rx)^2 + (SensPos(is1,2)- ry)^2);
    R2= sqrt((SensPos(is2,1)- rx)^2 + (SensPos(is2,2)- ry)^2);

    if R1<SensPos(is1,3)&& R2<SensPos(is2,3) % innenfor begges rekkevidde?
        ilp= ilp+1;
        % Invertert målestøymatrise
        %Pwi(ip,ip)= 1/(stdSFV)^2; %
        Pwi(ilp,ilp)= 1/(stdSFV*1.4142)^2; % (*1.4142 pga summen av to
målinger)

        % Skannfasemåling uten pertubasjon
        [z0]= mSFV(ip,SensPar,SensPos,rx,ry);

        % Skannfasemåling med pertubasjon
        [zdx]= mSFV(ip,SensPar,SensPos,rx+dx,ry);
        [zdy]= mSFV(ip,SensPar,SensPos,rx,ry+dy);

        D(ilp,1)= (zdx-z0)/dx;
        D(ilp,2)= (zdy-z0)/dy;
    end;
end; % ip=1:Npar

% Fiktive målinger for å unngå singulær kovariansmatrise
Pwi(ilp+1,ilp+1)= 1/(100e3)^2; Pwi(ilp+2,ilp+2)= 1/(100e3)^2;
D(ilp+1,1)= 1; D(ilp+2,2)= 1; % posisjonsmåling nord og øst

% Fisher's informasjonsmatrise som den inverse kovariansmatrise
FIM = D'*Pwi*D;
Pin= FIM;
%if cond(Pin)> 1000000
%   %disp(Pin);
%   inv22=Pin(2,2);
%   inv23=Pin(2,3);
```

```
%   inv32=Pin(3,2);
%   inv33=Pin(3,3);
%   fprintf( 1, 'inv22= %e inv23= %e inv32= %e inv33= %e
\n',inv22,inv23,inv32,inv33);
%   fprintf( 1, 'xs= %f ys= %f xfly= %f yfly= %f \n',xs,ys,xfly,yfly);
%end;

P=inv(Pin); %  Full kovariansmatrise

return
```

Function LFunk

```
function kvavvik= LFunk(x)
% Beregning av en skalar funksjon for ML-estimering
% TSm 30/5-2013 (fra tidligere, sist 23/8-2011)

% Funksjonen er sum kvadratavvik for alle målingene

global SensPar SensPos SFVm;

posE=x(1); posN=x(2);

[Npar,d1]= size(SensPar);
SFV= zeros(Npar,1); % målevektor

for ip=1:Npar,
    SFV(ip)= mSFV(ip,SensPar,SensPos,posE,posN);
end;

kvavvik= 0;
for ip=1:Npar, kvavvik= kvavvik + (SFV(ip)-SFVm(ip))^2; end;

return
```

Function CEPfraKov0

```
function [CEP,iunder]= CEPfraKov0(P)
% FUNKSJON FOR BEREGNING AV CEP FRA EN KOVARIANSMATRISE
%
% TSm 5/10-99 fra CEP_Sigm
%
% Setter inn funksjonsverdier i funk(2,..,6) som funnet i plott
% fra "CEPfunk", funk(1) gjelder for en en-dimensjonal fordeling.

Npkt=6;
Funk= zeros(1,Npkt);        % Funksjonsverdiene som skal interpolerers
Funk(1)= 0.675;
Funk(2)= 0.703;
Funk(3)= 0.807;
Funk(4)= 0.933; % før (feil?) Funk(4)= 0.893;
Funk(5)= 1.058;
Funk(6)= 1.178;
```

```
Funk(7)= 1.178; % forsøk på å unngå feil med indeks utenfor (CEP-formel
nedenfor)

P2= zeros(2);   % 2 x 2 kovariansmatrise for nord- og øst-posisjon
P2(1,1)= P(1,1);
P2(1,2)= P(1,2);
P2(2,1)= P(2,1);
P2(2,2)= P(2,2);

ev= eig(P2);
a= real(ev(1));
if a>real(ev(2)) b=real(ev(2));
else
   a= real(ev(2)); b=real(ev(1));
end;

if abs(a)<0.0001 a=0.0001; end; % (Fikk tilfelle med div med 0 i neste
statement)
rel= real(sqrt(abs(b/a)));
if rel==1 rel=0.99999; end;

iunder= real(fix(rel*(Npkt-1))+1);
iover= iunder + 1;
CEPrel= Funk(iunder) + (((rel*(Npkt-1))+1)-iunder)*(Funk(iover)-
Funk(iunder));
CEP= sqrt(a)*CEPrel;    % faktor i forhold til den største egenvektoren
return
```

Function mSFV

```
function [sfv]= mSFV(ip,SensPar,SensPos,rx,ry)
% Skannfase fra en radar mot et sensorpar
% TSm, 11/1-2013

Is1= SensPar(ip,1); Is2= SensPar(ip,2);
x1= SensPos(Is1,1); x2= SensPos(Is2,1);
y1= SensPos(Is1,2); y2= SensPos(Is2,2);

[sfv]= mSFV1(x1,y1,x2,y2,rx,ry);

return
```

Function mSFV1

```
function [sfv]= mSFV1(x1,y1,x2,y2,rx,ry)
% Skannfasevinkel (fra en radar mot et sensorpar); blir mellom +/-180
% TSm, 11/1-2013
%
% Koordinatsystemet flyttes slik at radaren kommer i origo og roteres til
% radar og sensor 2 ligger langs x-aksen. Vinkelen er da mellom x-aksen
% og ny posisjon for sensor 1

c1=180/pi; % omregning mellom radianer og grader
```

```
rot= atan2((y2-ry),(x2-rx));
x1ny= cos(rot)*(x1-rx) + sin(rot)*(y1-ry);
y1ny= -sin(rot)*(x1-rx) + cos(rot)*(y1-ry);
sfv= c1*atan2(y1ny,x1ny);

return
```