ABSTRACT
Synthetic aperture radar (SAR) is a digital signal processing technique which enhances the azimuth resolution of a radar image using the target doppler history created by the motion of the radar platform. If the platform deviates from a constant velocity, straight-line path then image quality is lost and image details become unfocused. Motion compensation (MOCO) is a technique in which the position and attitude of the platform is recorded or estimated and then used to correct the scene’s doppler history as if a straight-line, constant velocity path had been taken. Brigham Young University’s interferometric synthetic aperture radar (YINSAR) was flown on a Cessna Skymaster which experienced significant motion due to the aircraft’s small frame. But using multiple motion sensors, such as an intertial measurement device and various GPS units, the motion can be compensated for. This report discusses some basic SAR theory, SAR motion compensation, a brief description of YINSAR and measurement devices, the MOCO algorithm, and some radar image results.

SYMBOLS
\[ \delta x = \text{range resolution} \]
\[ c = \text{speed of light} \]
\[ B = \text{radar transmit pulse bandwidth} \]
\[ \lambda = \text{carrier frequency wavelength} \]
\[ D = \text{antenna size (in wavelengths)} \]
\[ \beta = \text{antenna azimuth beamwidth} \]
\[ \phi = \text{received pulse phase} \]
\[ R' = \text{range to nominal track/path} \]
\[ R = \text{range to actual track/path} \]
\[ \Delta R = \text{difference in range} \]
\[ \Delta \phi = \text{difference in phase} \]
\[ PRF = \text{pulse repetition frequency} \]

INTRODUCTION
In recent years, SAR imaging has become a viable solution to a wide variety of problems in areas such as archaeology, topographical mapping, environmental monitoring, and military situations. In order to be useful, the radar images must be accurate. However, conventional algorithms for the creation of SAR images assume that the radar travels on a straight-line path and is moving at a constant velocity. Unfortunately, when these requirements are not met the result is a defocused and deformed image. If radar platform motion is accurately measured then post-processing corrections can be applied to recover image quality.

SAR BACKGROUND
The resolution of a radar image pixel can be broken into the resolution of the range and azimuth (cross-range) directions (see Fig. 1). Resolution in the range direction is directly affected by the bandwidth of the radar transmit pulse and is given by

\[ \delta x \approx \frac{c}{2B} \]  

where \( c \) is the speed of light and \( B \) is the transmit pulse bandwidth. A higher bandwidth implies a shorter pulse length; however, if the transmit pulse is shortened, the received echo’s signal-to-noise ratio (SNR) also drops, resulting in noisy received echos and a noisier image. To overcome the balance between bandwidth and SNR, a linear frequency modulated (LFM) chirp pulse shape is used. When a pulse of this shape is correlated with another time-delayed copy, the result is a pulse with both a short duration (high bandwidth) and a high SNR (low noise). This process of creating higher range resolution by correlating the received pulse with the transmit pulse is known as range compression or range matched filtering.

In the azimuth direction, however, the pulse shape has no effect. The resolution here is directly related to the size of the antenna beamwidth which is directly related to the antenna size by the equation:

\[ \beta \approx \frac{\lambda}{D} \]  

where \( \beta \) is the antenna beamwidth, \( \lambda \) is the carrier wavelength, and \( D \) is the antenna size in wavelengths. The wider the beamwidth the poorer the azimuth resolution while the narrower the beamwidth, the better the resolution. But having a large antenna on a small airplane is
not realizable, thus a different solution is needed. SAR is a signal processing technique that uses many measurements from a small antenna to synthesize a large antenna, subsequently creating higher azimuth resolution. (see Fig. 1) The idea behind SAR is the tracking of the azimuth phase history of a target. If a constant velocity, straight-line path is maintained by the radar as it passes a target, the target’s collected phase history will be approximately quadratic. The significance of having a quadratic target phase history means that pulses received by the target will be shifted linearly in frequency as the aircraft flies by the target. This linear frequency shift in the received data is also known as a Doppler shift. With the phase history now viewed as a linear frequency shift or a LFM chirp, the phase data can be correlated in the azimuth direction as done for the range. The technique of increasing azimuth resolution using the SAR technique is also known as azimuth compression or azimuth matched filtering.

But in many cases maintaining a constant velocity, straight-line path is not possible and motion compensation of the data is necessary. YINSAR experiences many motion errors due to wind gusts and piloting errors, thus creating defocused radar images. These motion errors are measured by multiple sensors and then received data is corrected.

YINSAR

YINSAR is an X-band interferometric synthetic aperture radar created at Brigham Young University. The purpose of YINSAR is to demonstrate that accurate, high resolution SAR images can be generated using low-cost hardware. The radar platform is a four-passenger, dual-engine Cessna 337M owned and maintained by Utah State University (Fig. 2). The SAR operates at an altitude of 300m and has theoretical resolution of about a meter in both range and azimuth directions. More YIN-
### Table 1: YINSAR motion data collection devices

<table>
<thead>
<tr>
<th>Device</th>
<th>Collection rate(Hz)</th>
<th>Measurement format</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ashtech</td>
<td>10</td>
<td>Lat/Lon (WGS84)</td>
</tr>
<tr>
<td>Satloc</td>
<td>5</td>
<td>Lat/Lon (WGS84)</td>
</tr>
<tr>
<td>Trimble</td>
<td>5</td>
<td>roll, pitch, yaw in CW from N</td>
</tr>
<tr>
<td>INU Gyros</td>
<td>1000</td>
<td>Angle measurements taken from inertial axes</td>
</tr>
<tr>
<td>INU Accelerometers</td>
<td>500</td>
<td>Force measurements taken along inertial axes</td>
</tr>
</tbody>
</table>

into neighboring pixels, causing image inaccuracy. The second effect that translational motion errors have on the received data is a phase change in the received pulses. Viewing the path which a radar echo travels as a transmission line, the phase of the received pulse is given by

\[ \phi = \frac{4\pi R}{\lambda} \]  

(3)

Any deviations of the aircraft from its ideal path can be seen as a change in the transmission line length and consequently a change in the received pulse phase. Any change of this type will corrupt the phase history and subsequently defocus the image.

![Figure 3: MOCO geometry](image)

The other type of motion error presented here is error in azimuth spacing. In order for the phase history to be accurate, the spacing at which the radar pulses are collected must be uniform. This constant spacing is difficult to maintain because of varying cross range platform velocity. This error can be viewed as a making a synthetic array of antennas, but with unequal spacing between elements. The result of this type of error is also corruption in the phase history, which in turn defocuses the image and creates image inaccuracies.

### MOCO ALGORITHM

The YINSAR motion compensation algorithm consists of collecting aircraft attitude, position, and velocity, and then implementing corrections for translational motion and azimuth spacing errors. The algorithm involves three steps: a phase correction, a range bin interpolation, and an azimuth bin interpolation.

#### Phase Correction

As previously explained, aircraft deviations away from an ideal path introduces a change in the phase of received pulses. This change in phase, \( \Delta \phi \), can be represented by the change in range between ideal and actual paths, \( R - R' = \Delta R \), is

\[ \Delta \phi = \frac{4\pi \Delta R}{\lambda} \]  

(4)

To compensate for this phase variation, the range compressed SAR data, \( A e^{j\phi(p)} \), is simply multiplied by a phase correction term, \( e^{(j\Delta \phi)} \) and a motion compensated phase, \( e^{(j\phi_{corrected})} \)

\[ A e^{j\phi_{corrected}} = A e^{j(\phi_{p} + \Delta \phi)} = A e^{j\phi_{p}} e^{j\Delta \phi} \]  

(5)

#### Range Interpolation

As noted before, the other effect that range delay variations can have is a drifting of range bins. If a target’s returns are spread across multiple range bins, then the target appears smeared in the range direction in the radar image. These range bin inaccuracies can be corrected by interpolating each range line to an equal range bin spacing. To maintain phase and magnitude image accuracy, cubic spline interpolation is used.

In both the phase correction and range interpolation, the change in range, \( \Delta R \), caused by the deviations of the aircraft from an ideal track is needed. But in order to have accurate measurements of this range change, the exact positions of the aircraft and target are needed. Since target position is not known readily, it is estimated using the height of the aircraft above the ground.

#### Azimuth Interpolation

The last error correction performed by YINSAR’s MOCO algorithm is azimuth sample spacing correction.
When a radar platform’s velocity is constant, the azimuth sample spacing of a target’s returns is uniform. But, since a fixed pulse repetition frequency (PRF) is used, the azimuth sample spacing changes if platform’s velocity changes. Non-uniformity in along-track spacing degrades the synthetic aperture and makes suboptimal azimuth compression of targets. This motion error is corrected by interpolating the radar data in the cross-range direction to have an uniform azimuth spacing of \( \frac{1}{PRF} \). Cubic spline interpolation is used for higher accuracy in this correction.

**Implementation**

The initial implementation of the motion compensation algorithm was done in Matlab for simplicity of programming, and ease of debugging. The disadvantage to this implementation is the slow speed at which the Matlab script executes. The scripts were later converted to C for faster execution. Where possible, all C data structures mimicked Matlab structures and outputs of the C source were compared to Matlab source outputs throughout every step of the algorithm to ensure accurate behavior.

Two additional changes were applied during the translation. The first change was in interpolation schemes. Many times the Matlab source used linear interpolation rather than cubic spline interpolation to achieve faster processing times. The C source replaces these schemes with cubic spline interpolation for increased computation speed and better accuracy in interpolated phase and magnitude. The second change was in the lat/lon to UTM conversion function. The original function implemented in Matlab used an inaccurate approximation of the lat/lon to UTM conversion, moved a position anywhere from 100m to 4km from its original location during conversion. To solve this problem, a lat/lon to UTM conversion utility was borrowed from Jeeps, an open source mapping software, and accurate UTM positions were obtained from the converted lat/lon data.

**RESULTS**

In some cases targets were better focused in the motion compensated images (Figs. 4, 5). In other cases, images were found to have mixed results (Figs. 7, 6). Possible reasons that the motion compensation did not work as well as predicted were the inaccurate estimates of platform height above the ground (used in determining \( \Delta R \)), ignoring the other aspects of the motion compensation (accounting for the roll and yaw of the aircraft), and simplicity of the motion compensation (only applying first order motion compensation). Future work will include the estimation and correction of yaw and roll errors and implementing second order motion compensation.

**References**


Figure 4: BYU campus - without motion compensation

Figure 5: BYU campus - with motion compensation. Note the improvement in the detail in which artifacts in the image can be resolved (cars, buildings, etc.).

Figure 6: Hills outside Logan, UT - without motion compensation

Figure 7: Hills outside Logan, UT - with motion compensation. Note the trail beneath the two dots is better resolved, yet the dots themselves are less focused.