A SIMULATION MODEL OF THE DETECTION PATTERNS OF BIRDS USING MARINE RADARS

by

Christopher M. Souliere

A thesis submitted to the Faculty of Graduate and Postdoctoral Affairs
in partial fulfillment of the requirements for the degree of

Master of Science

in

Biology

Carleton University
Ottawa, Ontario

ABSTRACT

Many bird radar studies provide estimates of the number of birds flying past a given area, but very few of these actually estimate detectability. One of the challenges in radar ornithology is estimating the probability of detection of flying targets with altitude and distance**. I** estimated the detection patterns associated with three marine radars by using a combination of field trials and simulation modelling, and estimated the probabilities of correctly and incorrectly detecting birds in relation to altitude. Field trials were conducted between December 18th 2015 and January 08th 2016 in Russell, Ontario, using radars to detect an aluminum sphere suspended below weather balloons. The results indicate considerable variation in power among radar units. The nominal beam width (+/- 3dB) was 4 degrees and effective beam width as measured by the blip size was 7 degrees. Models of beam shape and return echo strength from these trials were implemented into a simulation model. The results indicate detectability varies with altitude, with few birds detected in the lower altitude bands relative to the highest. Many simulated birds were classified as two different birds when crossing the beam twice. The algorithm to remove background clutter was only partially successful, as there were many false detections, especially in the lowest altitude bands near the radar. Future bird radar studies should include data on detection probabilities, using the approaches described here or other comparable approaches.

ACKNOWLEDGMENTS

I would like to express my sincere gratitude to my supervisor Dr. Charles M. Francis for his guidance and unwavering support during this endeavor, for his patience in trying times and for sharing his wealth of knowledge.

A special thanks goes to my committee members for their insightful comments and generous feedback, especially so given the topic of my thesis. I thank Dr. Lenore Fahrig, Dr. Scott Findlay and Dr. Joseph M. Bennett.

My sincere appreciation goes to Dr. Philip D. Taylor and John M. Brzustowski for their invaluable advice in helping me structure the framework of my thesis and offering practical solutions to radar and simulation technicalities.

I am thankful to Lucy Meng, Thomas Villeneuve and Holly Stemberger for all the wonderful times we had in the lab – it was certainly more enjoyable with all the laughs we had.

In particular, I thank Thomas Villeneuve for assistance during the field season. I would also like to thank the folks at Russell Township for offering access to the Russell Lagoons to conduct my field trials.

Lastly, I must express my deepest thanks to my family and friends for their words of encouragement and support throughout this process.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF APPENDICES

INTRODUCTION

Birds were first detected by radars in the 1940s (Lack & Varley, 1945). Early studies primarily focused on the detection and identification of radar echoes as birds (e.g. Lack & Varley 1945; Edwards & Houghton, 1959). However, since the 1960s, radars have become a firmly established research tool in detecting and monitoring birds aloft, with most efforts focused on bird migration (Bruderer, 1997b). Since then, birds have been detected, monitored and quantified by various types of radars (Gauthreaux & Belger, 2003).

Radar is derived from the words RAdio Detection And Ranging; i.e. radio location, and is a technique primarily used to detect the location of objects in the atmosphere (Bruderer, 1997a). Radars operate by radiating radio waves in the form of a highly directional beam, which is often emitted as pulses to avoid interference between the transmitted and received signals (Radar Navigation, 2001; Griffiths *et al.* 2014). A portion of these pulses are scattered once they reach an object that has a dielectric constant different from the previously encountered medium, and a small portion of that scattered energy is reflected back to the radar (Bruderer, 1997a). Once the reflected energy is received by the radar, the weak signal is amplified and reproduced into video pulses that can display the desired information. To avoid interference and potential damage of the sensitive radar components from the transmitted and received signal, an electronic switch is applied to rapidly switch between states (Radar Navigation, 2001).

Bruderer (2003) provided an assessment of the various radars used for ornithological research and categorized them based on beam shape: either fan-beam (open array antenna) or pencil-beam (conical beam antenna). Fan-beam surveillance radars (e.g. airport-surveillance radars and ship navigation radars) produce beams wide in the vertical plane (e.g. 10-30°) and

narrow in the horizontal plane (≤ 2°). These radars offer high horizontal resolution, but virtually no altitude information, and have prominent ground clutter interference. Conical beam radars (e.g. weather surveillance radars, tracking radars and modified marine radars) offer a three dimensional resolution of targets in the atmosphere, reduced ground clutter and altitudinal information on targets. Low-cost ship navigation surveillance radars can be modified into conical-beam portable marine surveillance radars by retrofitting them with parabolic antennas. A tilter unit can be used to allow for conical scanning at different elevation angles (Figure 1).

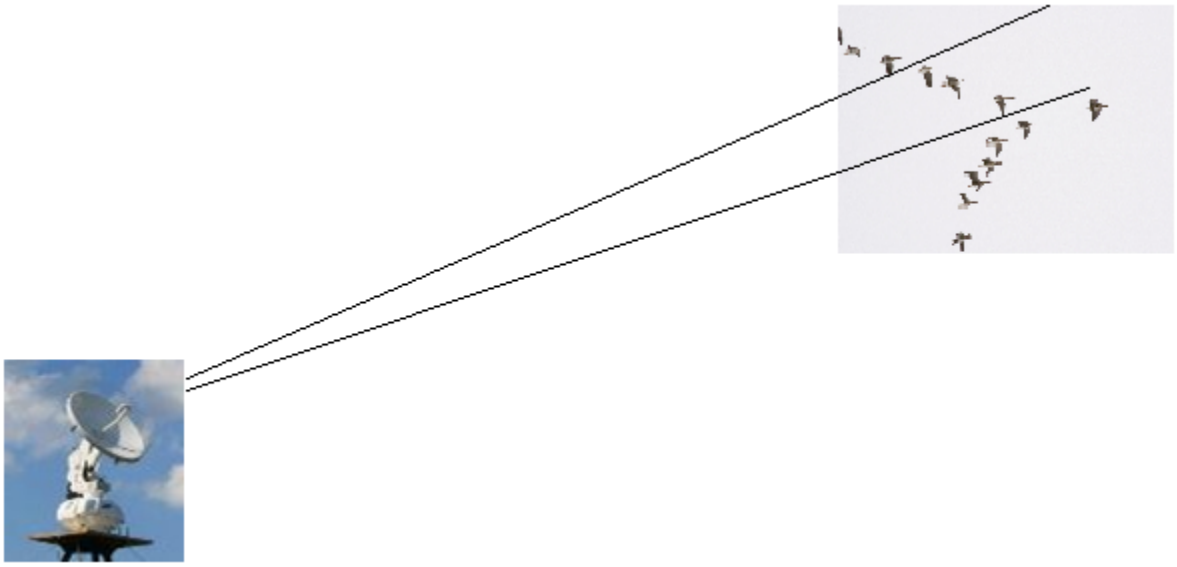

Figure 1: Example of a marine radar fitted with a parabolic antenna and using conical scanning. The elevation angles can be adjusted through a tilter unit that is software controlled.

Marine radars are well-suited for monitoring local bird movements. They provide information on the spatial distribution of birds, have high resolution, are relatively inexpensive and easy to maintain, are relatively compact, and are typically short-range (0-5km) (Harmata *et al.* 1999; Bruderer, 1997a; Williams, 1984). As a result, altitudinal information, flight directions and densities of birds can be locally quantified (Schmaljohann *et al.* 2008). Most ornithological

studies that use marine radars either operate open array antennas, parabolic antennas, or a combination thereof. Because of these attributes, marine radars are useful in ornithological studies that examine nocturnal migration at low altitudes, contrast between seasons and are situated in more remote areas (Williams, 1984).

The application of marine radars to ornithological studies has been ongoing since the latter half of the 20th century. One of the first studies using marine radars was conducted in the Mediterranean to assess bird migration (Casement, 1966). Recently, applications have expanded to include the influence of anthropogenic structures on bird movements, such as: wind turbines (Mabee *et al.* 2006), transmission lines (Deng & Frederick, 2001) and petroleum platforms (Day *et al.* 2005). Research applications include: estimating movements and numbers of threatened species (e.g. marbled murrelets; Burger, 2001); identifying dense migratory areas (Desholm *et al.* 2014); studying effects of weather on migratory behavior (Kahlert *et al.* 2012) and passage rates (Thomas *et al.* 2011); assessing theoretical flight speeds (Cabrera-Cruz *et al.* 2013); and understanding the influence of topography on migration (Williams *et al.* 2001).

One of the challenges of using radars to detect flying birds is estimating the probability of detection and how it varies with altitude and distance from the radar. Potential issues include variation among radar units, changes within a radar unit over time and differentiating birds from background clutter. Accurate estimation of bird densities requires understanding detectability. Many studies provide estimates of numbers of birds moving past a given area (e.g. birds migrating past wind turbines), but very few of these actually estimate detectability. Results from such studies may not be reliable without taking into account the detection probabilities of radars. Schmaljohann *et al.* (2008) outlined details on how to estimate the

detection probabilities of different echo sizes by calibrating a radar with a microwave signal generator. Dokter *et al.* (2013) used line-transects surveyed by field observers to quantify the detection probabilities of birds flying past a tracking radar, while utilizing proprietary tracker and clutter suppression algorithms. These two studies represent some of the few times in which there have been attempts to address the detectability issues encountered in bird radar studies. This, therefore, underscores the need for practical methods that provide ways to address the known capabilities and limitations of radars typically used in bird detection.

We are not aware of any studies that quantified detection patterns in ways that can be used to estimate the detection probabilities in the types of small marine radars that are frequently used for ornithological studies. There are several factors that need to be considered when evaluating the likelihood that a bird will be detected and identified by a radar (*sensu* marine radars). First, detectability varies with distance from the radar (birds very close to the radar may be less likely to be recognized as they cross the beam very quickly, while birds too far from the radar return echoes too faint to be recognized). Second, detectability varies with altitude, as birds too far above or below the centre of the radar beam are less likely to be detected. Third, the probability of detection may vary among radar units if the output power, internal components (e.g. the receiver) or other processing parameters differ.

An additional consideration of detection by radar is that some birds may be incorrectly classified when using standard radar processing algorithms, which can lead to potential sources of error. Some birds may fail to be detected when present, and some targets may be identified as birds that are not actually biological targets. Some of the potential sources of error include failing to detect a bird that is not in the beam, failing to detect a bird because it is filtered out

with the background environment, failing to recognize targets as birds through a 'tracker' algorithm, incorrectly identifying a single bird more than once, and incorrectly identifying background noise as birds. When processing radar data, background clutter may be classified as either static (e.g. buildings) or moving (e.g. leaves, waves). This leads to the need for algorithms that remove background clutter but do not remove birds, especially in the presence of fluctuating background noise. Because of these sources of error, a trade-off is required when using processing algorithms to minimize the risk of both failing to detect actual birds, and incorrectly identifying non-targets as birds.

The overall objective of this study is to estimate the detection patterns associated with modified marine radars by using a combination of field trials and simulation modelling and applying these to a bird dataset. The first objective was to estimate the actual radar beam shape and return echo strength at various distances empirically, using field trials with objects of known radar cross-section. The second objective was to develop a simulation model that incorporates the actual beam shape and return echo strength to estimate the detection probabilities of birds flying past the radar at different altitudes and distances using a standard set of processing algorithms. The third objective was to apply these detection probabilities to an existing migratory bird dataset to estimate the true altitude distribution of flying birds, and how this differs from the uncorrected estimate of the numbers of individuals detected in each altitude band. The approaches developed in this study could be relevant to all ornithological studies using marine radars to address biases associated with detectability.

METHODS

Objective 1: Estimate Beam Shape and Return Echo Strength

I conducted field trials with three radars and aluminum spheres of known radar cross-section suspended below weather balloons to estimate beam shape and return echo strength empirically, based on recorded digitized echo patterns for the spheres. I used these data to develop models of the relationship between digitized targets and distance from the radar and position in the beam.

Study Site and Radar Locations

I operated three modified portable marine X-band radars (two Furuno 1954C-BB and one Furuno 1964C-BB) during daylight hours intermittently between December 18th 2015 and January 08th 2016 to detect an aluminum sphere suspended below weather balloons. All radars were operated within the boundaries of the Russell wastewater treatment lagoons in the township of Russell, Ontario, Canada. I chose this site because of the lack of buildings and ground vegetation, which allowed for an unobstructed view of the atmosphere. The radars needed to have a clear line of site to the target object so that the beam did not become contaminated with clutter, and therefore affect the quality of the radar data. The site is characterized by five rectangular-shaped ponds and a triangular-shaped marsh demarcated by gravel service roads and bifurcated by a transmission tower and power lines (Figure 3). I deployed the radars adjacent to the ponds on the periphery of the site with as much distance between each unit as allowed by the layout of the site (Figure 2).

Figure 2: Map of the study site showing the locations of the three radars, named “Eagle”, “Bunting” and “Plover”. The approximate locations where spheres were raised to be detected on each of the radars are shown as follows: EM: Eagle medium; PM: Plover medium; PS: Plover small; BM: Bunting medium; BS: Bunting small. Medium and small denote sphere sizes.

Figure 3: Radar setup and the field site in early January. An example balloon trial setup from a previous attempt in the summer is shown on the right, with a sphere attached to a set of three weather balloons. The arrangement was attached to a spool via a low-stretch synthetic cord allowing the setup to be raised and lowered.

Radar Description

I used small X-band marine surveillance radars (~3.2 cm wavelength) retrofitted with a parabolic antenna (~60 cm diameter) which completed a full revolution (360° scan) every ~2.37 seconds (Table 1). The radars operated at a frequency of 9.41 GHz and had a nominal beam width of ~3.5°. The antennas of each radar were tilted to fixed angles to target the expected position in the atmosphere in which the three radar beams would intersect the sphere suspended below the weather balloons. All radars were manually tuned prior to field trials.

Table 1: Summary of the name, power, location, altitude and operational date of all three radars.

Radar Name	Peak Radar Power (kW)	Latitude	Longitude	ASL (metres)	Operation Date
Plover	25	45° 15' 30.960" N	75° 20' 00.216" W	72.1	a,b,c
Bunting	25	45° 15' 15.510" N	75° 19' 51.648" W	68.8	c
Eagle	12.5	45° 15' 33.084" N	75° 19' 20.532" W	70.7	b

[a] 16/12/2015
[b] 18/12/2015
[c] 08/01/2016

Digitization of the Radar Signal

The radar digitization process follows the methods outlined in Taylor *et al.* (2010). I used the open source program radR to acquire and analyze signals from each radar scan. The basic data structure of radR is a matrix of integers representing power received by the radar at a certain elevation angle, azimuth and distance, and discretized within a set of two-dimensional pixels (samples). Each column of the matrix represents a pulse of radio energy (radial dimension) reflected from the surveyed volume of space at increasing distance from the radar. Each row represents the energy received from a particular 'range cell' (angular dimension) away from the radar, as the radar is pointing at a given azimuth. Thus for a single pulse of energy, the intensity of a return echo is digitized into specified 'samples' corresponding to the volume of space occupied by the reflected object.

USRP Calibration

Analog radar signals were digitized using a modified Universal Software Radio Peripheral (USRP) device that can act as a high-quality marine radar digitizing card. The 12 bit cards store a sample value as an integer ranging from 0-4095 (i.e. 2^{12}-1) (Taylor *et al.* 2010).

The three USRP's used on our radars were calibrated by feeding microwave (Wavetek 907A-S-792 microwave signal generator) pulses of known power into a single radar (a Furuno FR 1965 X-band) using an ultraflex cable, a cable waveguide-adapter and a flexible waveguide, to 'funnel' the beam. For each USRP card, a function was developed empirically to map USRP values (intensity) to decibel-milliwatts (dBm) entering the waveguide. All values mapped to the range 2115-4095 USRP values. To implement the mapping, a calibration factor was calculated for a radio frequency (RF) probe at 9.41 GHz using a power meter (Marconi RF power meter 6960). Next, the output from the microwave generator was validated against an RF meter via the RF probe by implementing a linear model that checked whether the microwave signal generator output showed good linearity across corrected (with attenuator) power meter readings. The attenuator is designed to increase the dBm range in which the signal shows good linearity. Afterwards, data were read from the USRP and used to fine-tune the microwave signal generator frequency to maximize the peak value, which is intended to mimic the auto-tune function on a radar's own receiver. Finally, a nonparametric locally weighted polynomial regression (LOWESS) was used to smooth the measurements and map USRP values to dBm and vice versa for each USRP card.

Weather Balloon Trials

We used two different aluminum spheres of diameters 7.61 cm and 12.67 cm and suspended them below weather balloons filled with helium to obtain information on beam shape and return echo strength (Figure 4). These correspond approximately to the radar cross-section of a medium to large bird such as a duck (Vaughn, 1985). The radar cross-section is the measure of an object's ability to reflect radio waves and has the dimensions of an area (cm^2); it can be thought as the apparent size of an object as seen by particular radar (Bruderer, 1997a, Griffiths *et al.* 2014). We used a low-stretch synthetic string to suspend one sphere at a time below the weather balloons at a distance large enough that the balloons would not be in the beam at the same time as the sphere. We chose various spots along the gravel service roads in which we expected to raise the sphere through the radar beams to obtain multiple distance recordings. The weather balloons were attached to a low-stretch synthetic cord via a spool and incrementally elevated until the sphere intersected one or more radar beams. This was confirmed by an operator at the radar identifying the persistent shape and consistent echo strength of an object considered to be a sphere on the plan position indicator (PPI or radar display). At this point, the weather balloons remained stationary while the approximate centre of all three radar beams traversed the sphere. However, due to wind and limitations in where we could walk due to the ponds, the sphere was rarely at the centre of all three radar beams, and therefore, often detected by only a single radar. As a result, we moved the balloons into a suitable location where the approximate centre of a single radar beam could detect the sphere for a period of time (Figure 4). Once we estimated we had a large enough number of scans to get good quality measurements (i.e., ~300-500 measurements near the center of the beam) of

targets at each location, the sphere was moved to another location either further or closer to the radar along the gravel service roads. Once done, we switched sphere sizes and repeated the process with the same radar, again recording at multiple distances from the radar. Subsequently, we repeated the process anew with another radar, again switching sphere sizes. We attempted to record altitude information by attaching two altimeters to the weather balloons, but the altimeters failed to record accurate data. We recorded the time of launch and arrival for every sphere trial, and also recorded time as a marker when the sphere appeared on the radar display which was used for subsequent radar data processing.

The software radR can record either a blipmovie, which is partially processed to store samples of putative biological targets ('blips'), or a raw archive which stores all samples from each scan (Taylor et al. 2010). I recorded everything in raw during field trials and also collected data when there were no flying targets (e.g. birds or balloons) in the surrounding environment, so that they could be later merged with simulated targets to create a more realistic radar recording when testing the effectiveness of radar processing algorithms.

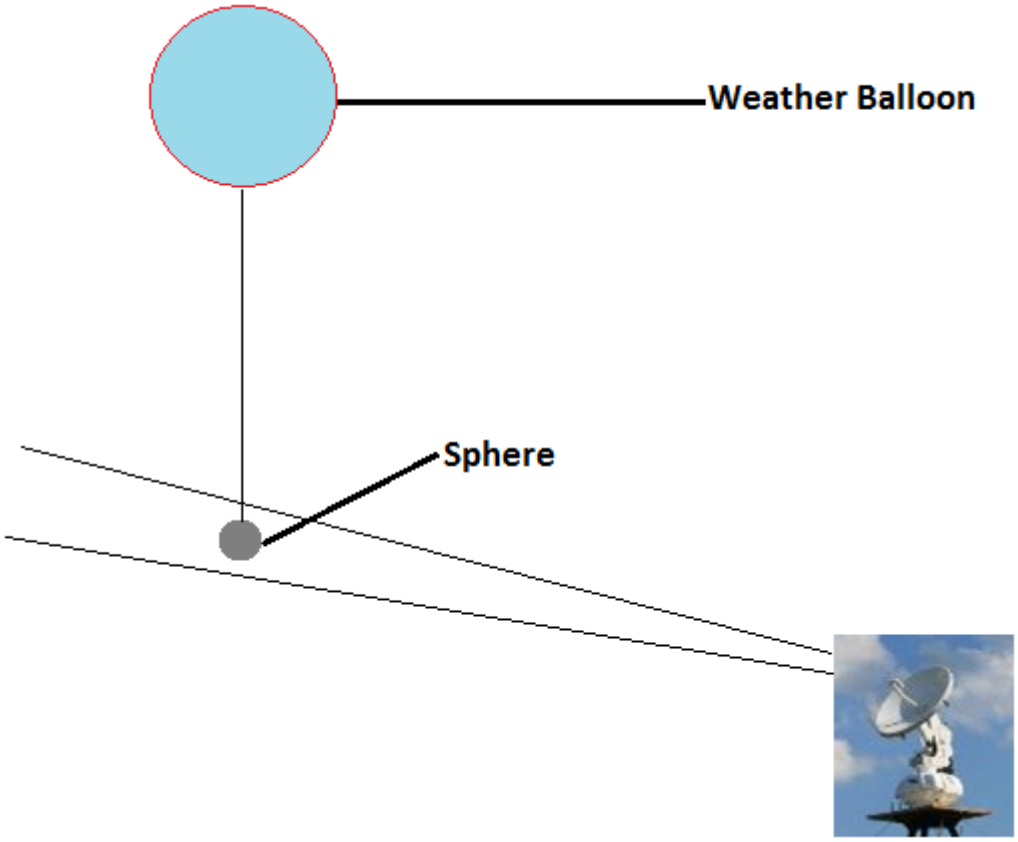

Figure 4: Schematic of the setup used for empirical field trials. The sphere was suspended below the weather balloon and raised or lowered with a spool (not shown) to attempt to get it into the centre of the radar beam.

Target Extraction

I extracted sphere targets from recorded raw movies using a target extraction script from radR and saved the data in a common R storage format. I manually located targets within the radR display based on several criteria: the persistent and symmetrical pattern and echo strength of the targets on the display; the number of scans in which the targets were visible; the position of the targets in relation to the estimated distance and angle of the balloon from the radar; and the times that the targets were visible in relation to when the balloon had been flown. I excluded targets manually (through examination of the target in radR) and

automatically (an R script that processed extracted targets) that were strongly asymmetrical (e.g. incompletely detected or contaminated by atmospheric anomalies or stray radar beams). I then used the extracted targets to determine the beam shape and loss in power as a function of distance.

Power Function

I took several steps to determine received power (i.e. return echo strength) as a function of distance for each sphere size. Firstly, to eliminate the inherent variation in the digitized-to-analog signal between radars, I used the mapped functions from the USRP calibration and applied them to the extracted sphere targets to convert their corresponding USRP values into adjusted dBm values. The adjusted dBm values were then converted into milliwatt/cm^2 using the following equation

$$\frac{mWatt}{cm^2} = 10000 * 10^{dBm/10} \quad (1)$$

Radar theory stipulates that received power declines to the fourth power of range, in which power units generally take on the form of Watts/m^2. I used the radar range equation to determine the return echo strength of the sphere targets as a function of distance from the radar. The radar range equation is as follows

$$P_e = \frac{P_t G^2 \lambda^2 \sigma}{(4\pi)^3 R^4}, \quad (2)$$

where

P_e = received power (W)
P_t = transmitted power (W)
G = antenna gain (dimensionless)
λ = wavelength (m)
σ = radar cross-section of the target (m^2)
R = range (m),

which simplifies to

$$P_e = \frac{C}{R^4} \,,$$

(3)

since P_t, G, λ, σ, and $(4\pi)^3$ are constant for each sphere size and dBm values are adjusted to compare amongst radars. Thereafter, I simplified equation (3) in order to fit a log-log non-linear least-squares regression (4) to the extracted sphere targets of Eagle and Plover together and Bunting separately using the medium sized sphere, with the expectation of empirically modelling the loss in power as a function of distance. I did not use the small sphere in the analysis as there were no recordings beyond 300 m. Since I failed to collect any altitude data on the sphere targets, I chose a sphere target that had the largest return echo strength (i.e. peak power) per range cell, on the assumption that the highest return echo strength per range cell corresponded to the centre of the radar beam. The general form of the power model is as follows

$$P_e = e^{ln(C) - (\alpha * \ln(R))} \,,$$

(4)

where α is the estimated power coefficient of the range. I chose to implement the Plover and Eagle model into the simulation because Bunting had a weak signal.

Beam Shape Function

Several steps were followed to determine the empirical beam shape. In order to model beam shape and implement the results into the simulation, I assumed that the beam was symmetrical in both the vertical and horizontal planes. The beam can be considered as a certain number of pulses (radial dimension or depth) successively hitting and simultaneously sweeping across the sphere target in a horizontal fashion. These pulses are digitized by the USRP's along the horizontal plane and collectively acquire a certain width. This calculated width is then subsequently used in the simulation as the parameter values that make up both the horizontal and vertical beam width. I chose an extracted blip with high relative power and symmetry compared to all other extracted blips to characterize the beam. Next, I used the chosen blip's adjusted dBm values (obtained from applying the USRP conversion functions) and back-converted them into USRP values. I graphed power in URSP values corresponding to each range bin against angle and developed an approximate model based on second-order polynomials that characterize the blip, with the purpose of simulating radar beam shape by using a target of known radar cross-section located at the approximate centre of the radar beam. All statistical analyses were performed in R (R Core Development Team, 2015).

Objective 2: Radar Simulation and Estimating the Probabilities of Detection using radR Algorithms

To estimate the probabilities of detection, I simulated birds crossing the radar at various altitudes using a modified version of the genblips module in radR. The size and shape of the blip created by a simulated bird whenever it crossed the beam was simulated to match that of the

real spheres, adjusted for the position in the beam and distance from the radar. I then evaluated the effect of commonly used radR processing levels to estimate the proportion of simulated targets that would be retained after processing and recognized as tracks. I subsequently merged a recording of the simulated targets with a background recording of a real radar environment and again evaluated the effect of commonly used radR processing levels.

Genblips Module

I simulated bird targets by modifying an existing module in radR called "genblips" to make the parameters and internal functions more realistic to match parameters calculated for spheres. Genblips is a module within radR that simulates targets starting at a position outside the user-defined range of the radar and moving in a random direction with a predefined speed, altitude and direction. Targets that move away from the radar are eliminated, while those that cross within the range of the radar are tracked and tabulated. The original version of genblips simply calculated whether or not the target crossed the beam, and created a fixed shape blip regardless of distance or position. I modified this to create realistic blips based on the actual shape of blips representing spheres, and adjusted in strength according to their position in the beam and distance from the radar.

I modified the genblips module by first specifying the dimensions of the artificial targets via the number of samples that made up the angular and radial dimensions of the sphere target. Since the radar cross-section of the medium sized sphere (~130 cm^2) is approximately equivalent to a large to medium sized bird (~100 cm^2; Vaughn, 1985), the dimensions that constitute the sphere target were used as the dimensions of the artificial targets in the

simulation model. I then 'cut' the modelled beam function into intervals according to the number of digitized samples in the angular dimension. These cuts (or sample intervals) took on power values (y-axis) according to their distance away (along the x-axis) from the axis of symmetry of the modelled beam shape. In other words, since beam shape was modelled with a quadratic function, the sample intervals that lay on the axis of symmetry had larger power values then the sample intervals further away. The vertex (the axis of symmetry) of the modelled beam shape represented the centre of the radar beam.

The modelled power determined the peak power of the artificial targets at a certain distance from the radar. The peak power always corresponded to the centre of the radar beam or vertex of the modelled beam shape. If an artificial target happened to be in the radar beam but at a certain distance from the centre, then its power value was lowered according to the modelled beam shape. This adjusted power then became the peak power of the artificial target and acquired the same shape as a target in the centre of the beam, unless the power became so low that the target was cut off by the background threshold. Since the sphere target used to model the beam shape was recorded at a distance from the radar, the peak power was relatively lower than that of a closer distance. For the simulation, I scaled the peak power and corresponding beam shape either upwards or downwards in power, when the artificial target was either closer to or further from the radar, than the chosen sphere target used to model beam shape. The scaling occurred irrespective of the position of the artificial target in the beam.

To summarize, according to the power function, the distance of an artificial target from the radar determined the peak power, assuming the target was at the centre of the radar

beam. Given a fixed distance, if an artificial target was not at the centre of the radar beam, then the peak power was adjusted based on the angle above or below the centre of the radar, based on the drop-off in power within a blip on the horizontal axis. That adjusted peak power and the corresponding beam shape were 'painted' onto the samples that form the artificial target along the angular dimension (width). The modelled beam shape (and hence the digitized artificial targets) had an apparent depth that corresponded to range cells away from the radar, representing echo strength from different parts of the same pulse. Each corresponding range bin (all range cells of the digitized target that were at the same distance from the radar) had a unique modelled beam shape as dictated by the digitization process. For each range bin, the corresponding modelled beam shape was 'painted' onto the set of range cells that constitute it. Samples (cells or pixels) from artificial targets that were below the atmospheric parameter threshold (i.e. not detected by the beam) were removed from the simulation. In this way, the artificial targets displayed in every scan were modelling power and beam shape according to their distance from the radar and relative position to the centre of the radar beam, respectively.

Merging Simulated Data with Real Data

To evaluate radar processing algorithms in a realistic environment, I merged the blipmovie produced from the genblips simulation with a real radar recording containing clutter and background noise, but no actual flying targets (i.e. no birds, bats or insects). I did this by merging the raw recording made at the field site with the simulated blipmovie. To do this, I took the maximum value occupying the same sample-pulse slot (i.e. pixel or sample) across both archives. The maximum value was chosen because when two objects occupy the same

sample-pulse slot, the largest echo strength will emanate from the object that has the greatest reflectivity potential. Given the raw recording had stray radar beams after 500 scans, I used a repetition of the first few hundred scans (approximately 460 scans), rather than the full original raw recording, when merging both archives. The merged raw archive was subsequently processed by radR into a blipmovie archive, which consistently eliminates stationary clutter (e.g., buildings, ground reflection), and groups echoes in adjacent cells into blips or targets (Taylor et al. 2010).

Processing Blipmovie Archive

Processing of a blipmovie archive mainly follows the methods outlined in Taylor *et al.* (2010). To detect targets in the environment, a background matrix is computed from multiple scans using user-defined polygons of pulses (radial dimension) and samples (angular dimension). The temporal mean and mean deviation of the radar signal echo strength is computed for each region of a scan. Subsequently, within each region of a scan, and each sample, the z-score (intensity of a signal return) is computed against the relative background distribution. If the z-score of a sample exceeds the background threshold, then the sample is considered 'hot' and grouped with adjacent hot samples into a patch. The patch can either be considered 'hot' or a blip if it further satisfies user-defined filtering criteria. The target finding algorithm is continuously updated to allow for changes in background clutter (e.g. changed due to adjusting the antenna angle).

Processing Levels

I applied 4 levels of processing to the results of the simulation model to look at the effectiveness of different processing scenarios typically used by radR for identifying putative

biological targets. The 4 processing levels are outlined in Table 2. Each level is roughly intended to increase the realism of the simulation of birds flying past a radar, apart from the first processing level, which simply counts simulated targets and tracks detected by the radar. The processing levels were applied to a simulated dataset and a simulated dataset merged with a realistic background.

Table 2: The four processing levels and their units, and description used in the two simulation steps.

Processing Level	**Units**	**Description**
None	Targets or Tracks	Analysis of targets and tracks generated from the simulation that are detected by the radar (i.e. in beam)
Blip	Targets	Applying blip filtering criteria that would be used for real radar data processing to a: 1. simulated dataset 2. simulated dataset merged with background
Blip and Clutter	Targets	Applying blip filtering criteria and a clutter map that would be used for real radar data processing to a: 2. simulated dataset merged with background
Tracker	Tracks	Applying tracker algorithm to: 1.blip processed dataset 2.blip and clutter processed dataset merged with background

Simulation Parameters

I applied user-defined parameters (Table 3) to the genblips module which generated a blipmovie that simulated bird targets flying past a radar according to the modelled power and beam shape functions. I started targets outside the max range of the radar beam and dropped them from the simulation once they were 1.6x outside the max range of the radar beam. I generated a random number of new targets per scan using a Poisson distribution ($\lambda = 5$). Starting positions, airspeed, altitude and azimuth were generated with a random uniform distribution within a set range. The airspeed, altitude and azimuth of each simulated target remained constant once it was generated. I chose an airspeed range based on findings from Cabrera-Cruz *et al.* (2013). I set the horizontal and vertical beam width using the modelled beam width. The module also generated a file which tracked blip information across every scan (e.g. track number, position, speed, peak power, etc.).

Table 3: Parameters used in the genblips simulation.

Beam Angle (deg.)	Max Range (m)	Air Speed (m/s)	Altitude (m)	Azimuth (deg.)	Scans	Atmospheric Threshold (minimum USRP value)
30	2561.17	7 - 20	0 - 1280.58	0 - 360	2500	2250

Blipmovie Processing Criteria

I processed the blipmovies using parameters that had been previously been used for processing radR blipmovies from actual migrating birds to remove clutter and detect birds (Villeneuve, 2015). This involved filtering blips based on the maximum and minimum of four user-defined filtering criteria (Table 4): the blip samples criteria which is the number of fixed

samples (pixel or sample cells) that can constitute a blip; the PPI area which is the apparent size of the blip in m^2 as seen on the radR display; the radial (depth) and angular (width) span criteria which measure the sample dimensions of the blip (Taylor et al. 2010).

Table 4: The minimum and maximum threshold values used as the filtering criteria to remove unwanted targets, and thereby generating more realistic bird targets.

Criteria	**Minimum**	**Maximum**
Blip Samples	30	5000
PPI Area (m^2)	100	5000
Angular Span (# samples)	3	-1*
Radial Span (# samples)	2	25

*denotes infinity in radR's blip processing parameters

Removing Clutter

A complication arises when a raw archive of a real radar recording is merged with a blipmovie, namely the problem of persistent but fluctuating clutter (e.g. fluttering leaves) that is retained from the real radar recording because it is harder to identify automatically than stationary clutter (Taylor *et al.* 2010). I worked around this complication by applying the 'declutter' plugin from radR, which removes persistent but fluctuating clutter by using an occupancy threshold applied to each sample slot within each scan.

Testing the Tracker Algorithm

To identify a moving target, blips from individual scans need to be concatenated across multiple scans into tracks. I tested the effectiveness of the multi-frame correspondence (MFC) algorithm (Shafique & Shah, 2005) used by the 'tracker' plugin in radR to create reproducible

tracks from blips that were generated by the genblips simulation with various levels of processing. Using this method, blips across multiple scans may be considered the same moving target if the blips adhere to certain algorithm constraints and user-defined criteria. The method begins by pairing blips from consecutive scans using the nearest neighbor approach and assigning a velocity to the matched blips. In the subsequent scan, the algorithm considers all possible blip matches from the first two scans to those of the third. Each new blip is matched to an existing track via a 'gain' function that assesses the quality of the match. The gain function optimizes between the proximity of the new blip to the location predicted by the track and directional consistency when the new blip is added to a track. The procedure is repeated for subsequent scans. The user-defined parameters 'gain' (minimum gain for a blip to join a track) and 'alpha' (weight of directional consistency vs. proximity to prediction) define whether a blip is added to a track. Both parameters range between 0 (poorly matched) and 1 (perfectly matched) (Taylor et al. 2010). I used a gain of 0.9843 and an alpha of 0.6 to match the settings used by previous researchers using the same software to process actual bird recordings (Villeneuve, 2015). I compared the tracks created by the 'tracker' algorithm with the actual number of tracks generated in the simulation to test the effectiveness of the tracker algorithm.

Objective 3: Applying the Probabilities of Detection to a Migratory Bird Dataset

I calculated the probabilities of detection (POD) of the standalone simulation and the blip-and-tracker processed simulation as the proportion of tracks that crossed the radar beam out of all targets that were in range of the radar for each altitude band. Both probabilities of detection were calculated from the simulation without the merged background. I applied both

sets of POD to a migratory bird dataset recorded between October 8th - 15th, 2013 in Essex County, Ontario, Canada. The dataset was processed prior to this study and closely followed the methodology outlined in Villeneuve, 2015.

RESULTS

Beam Shape and Power Models

A second-order function fitted relatively well to a symmetrical blip extracted from the center of the beam (Figure 5), providing a suitable model of beam shape to apply to the simulation (Range Bin 536 m: R^2 = 0.9877, Y = 3081.8734 + 28.2692*X – 30.5134*X^2; see appendix A for other models). I chose to analyze a blip from Eagle since it had good symmetry and high relative power, most likely indicating the center of the beam. The nominal beam width (+/- 3dB) was approximately 4 degrees and effective beam width as measured by the blip size was approximately 7 degrees.

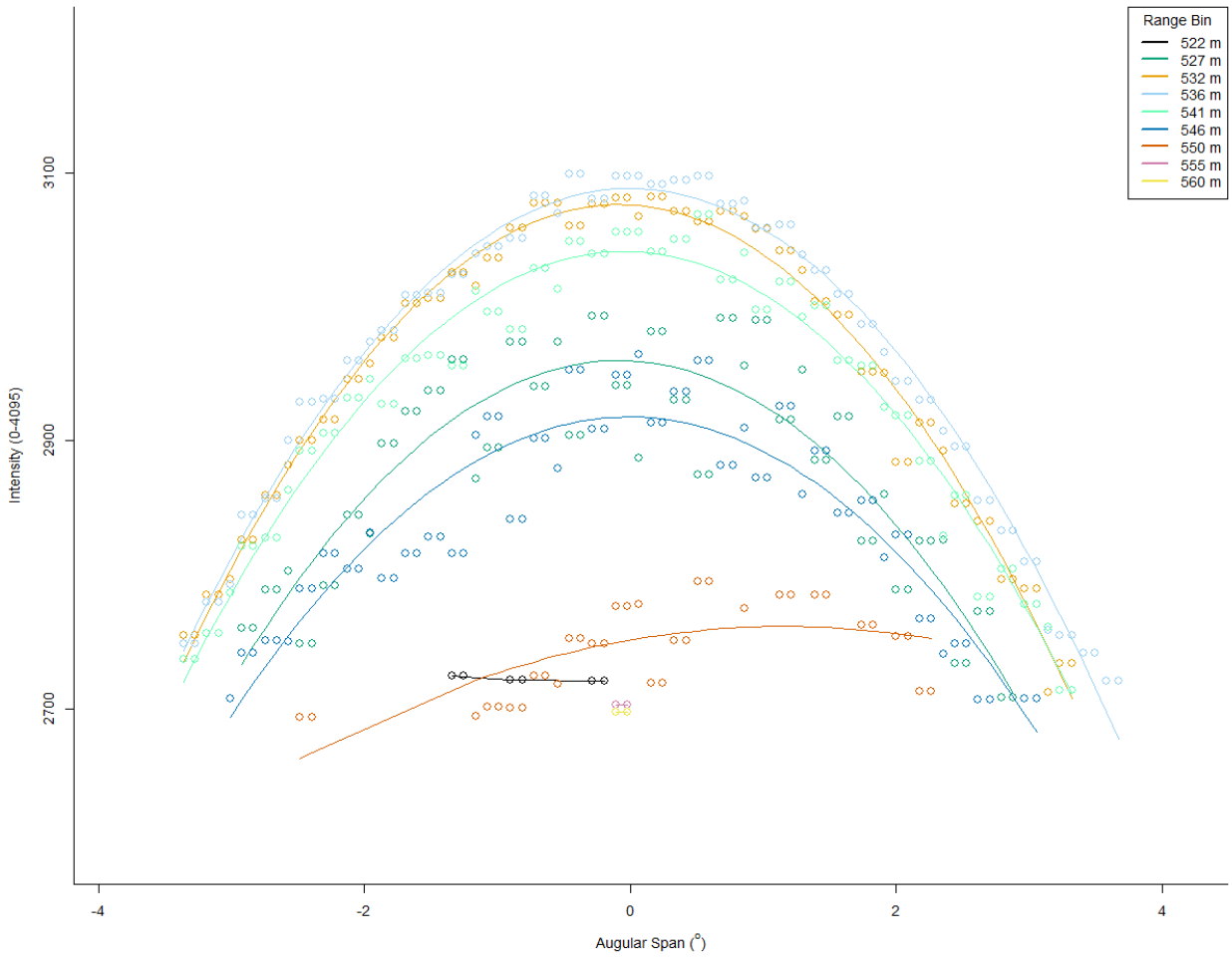

Figure 5: The empirical beam shape at the approximate centre of the radar beam; modelled using Eagle and a 12.67 cm diameter aluminum sphere. The y-axis shows the intensity scale as returned by the USRP. The range bins refer to the apparent surveyed volume occupied by the sphere, representing echo strength from different parts of the same pulse.

A fourth power function appeared to fit relatively well to the peak power within a range bin of blips from some radars, but not to all three radars combined. Only peak power per 4.68 m bin (range cell interval) was used in the models since peak power is assumed to correspond with the approximate centre of the radar beam (Figure 6). A fourth power function fitted well to Plover and Eagle (ln(Power Received) = 24.6781 – 4.02 ± 0.011 * ln(Range)) together and

Bunting (ln(Power Received) = 23.5958 – 4.01 ± 0.011 * ln(Range)) separately, with both functions providing a suitable fourth power model to apply to the simulation.

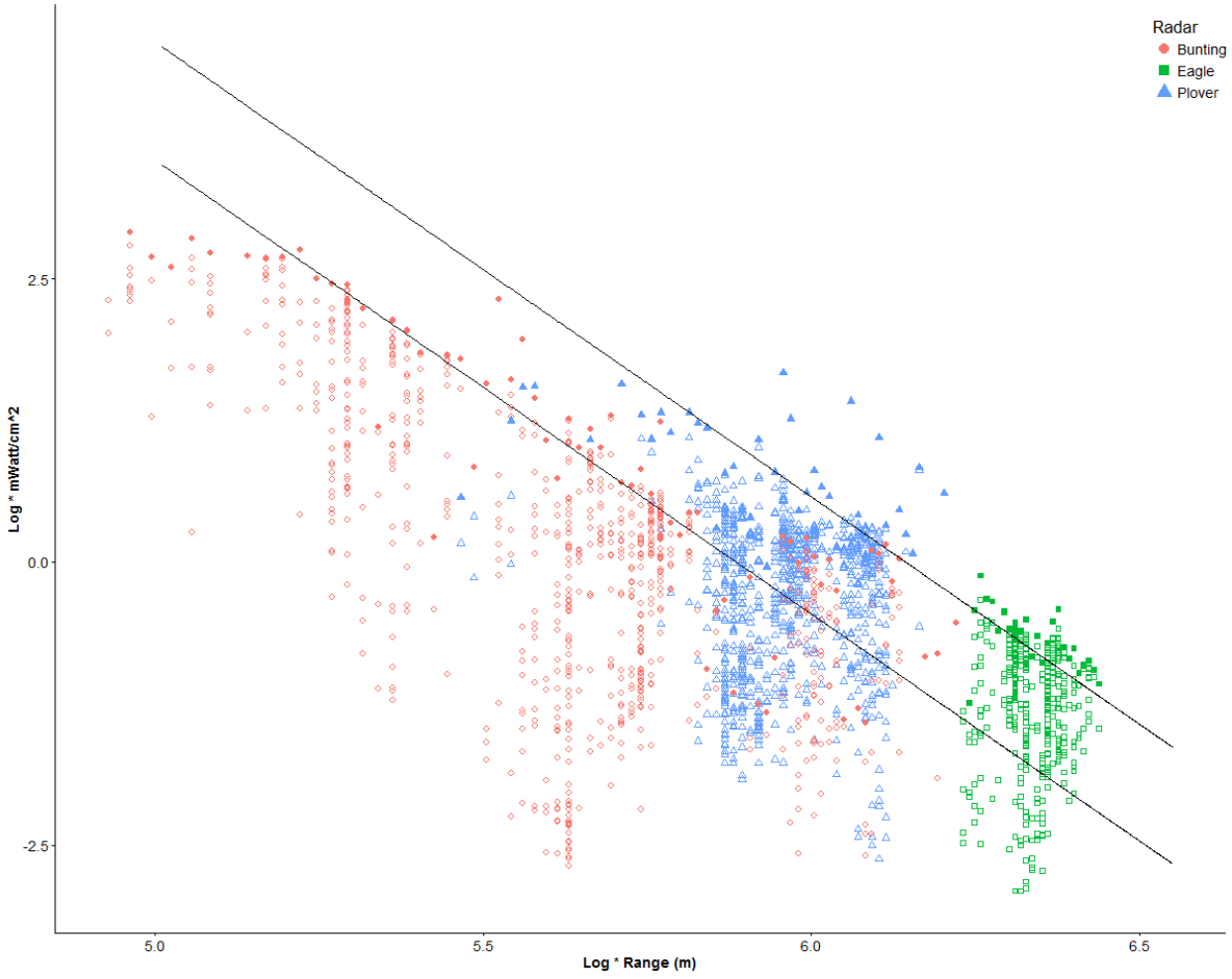

Figure 6: Loss in power as function of distance from the radar. Models fitted to peak values per range bin with Plover and Eagle pooled together and Bunting separate. The filled icons refer to the peak power values per range bin; the open icons refer to all other power values. Scaled with natural log.

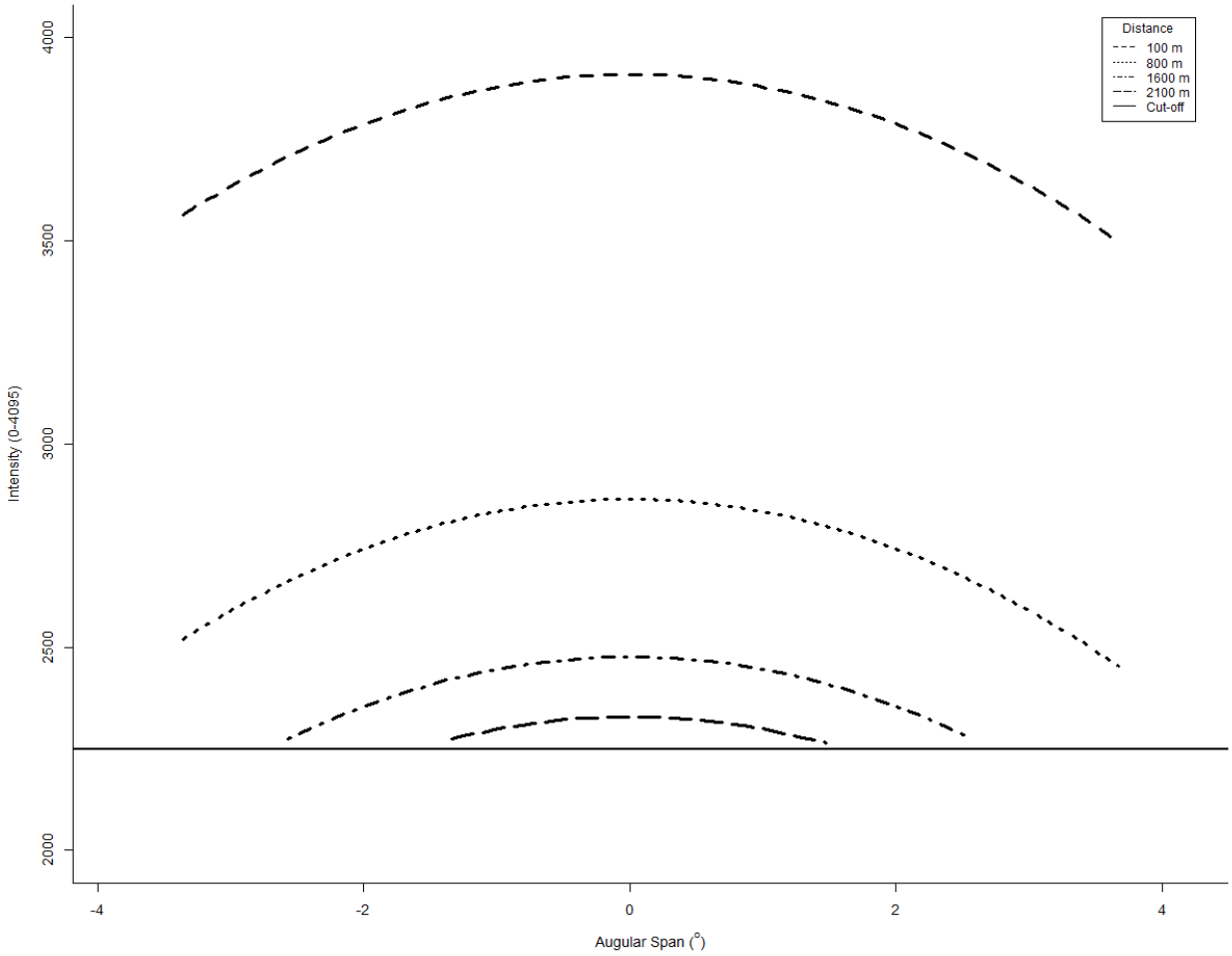

Figure 7: Example of the beam shape and power of a simulated target at four distances from the radar. The cutoff refers to a threshold value where certain samples of the simulated target cannot be distinguished from the atmosphere, and therefore are no longer in the beam.

Simulation and Processing

The beam shape and power models were implemented as parameters into a simulation consisting of 2500 scans and subsequently passed through various processing levels (see Figure 7 example). For simulation purposes, I chose to use the Eagle-Plover equation (ln(Power Received) = 24.6781 – 4.02 ± 0.011 * ln(Range)).

There were 3480 simulated tracks that came within range of the radar (2561.17 m) of which 1621 crossed the beam at least once generating a total of 37629 blips (Table 5). Overall targets were detected less than 10% of the time that they were within theoretical range of the radar, indicating they were often above or below the beam. Nearly 50% of simulated tracks were detected in the radar beam although the number dropped to slightly more than 40% after applying the blip processing and tracker algorithm. Most tracks were detected crossing the beam twice (e.g. Figure 8) and identified as separate tracks (77%), while only 21% were correctly identified as a single track. Roughly 2% were detected as three or four tracks, or misdetected altogether (Table 6).

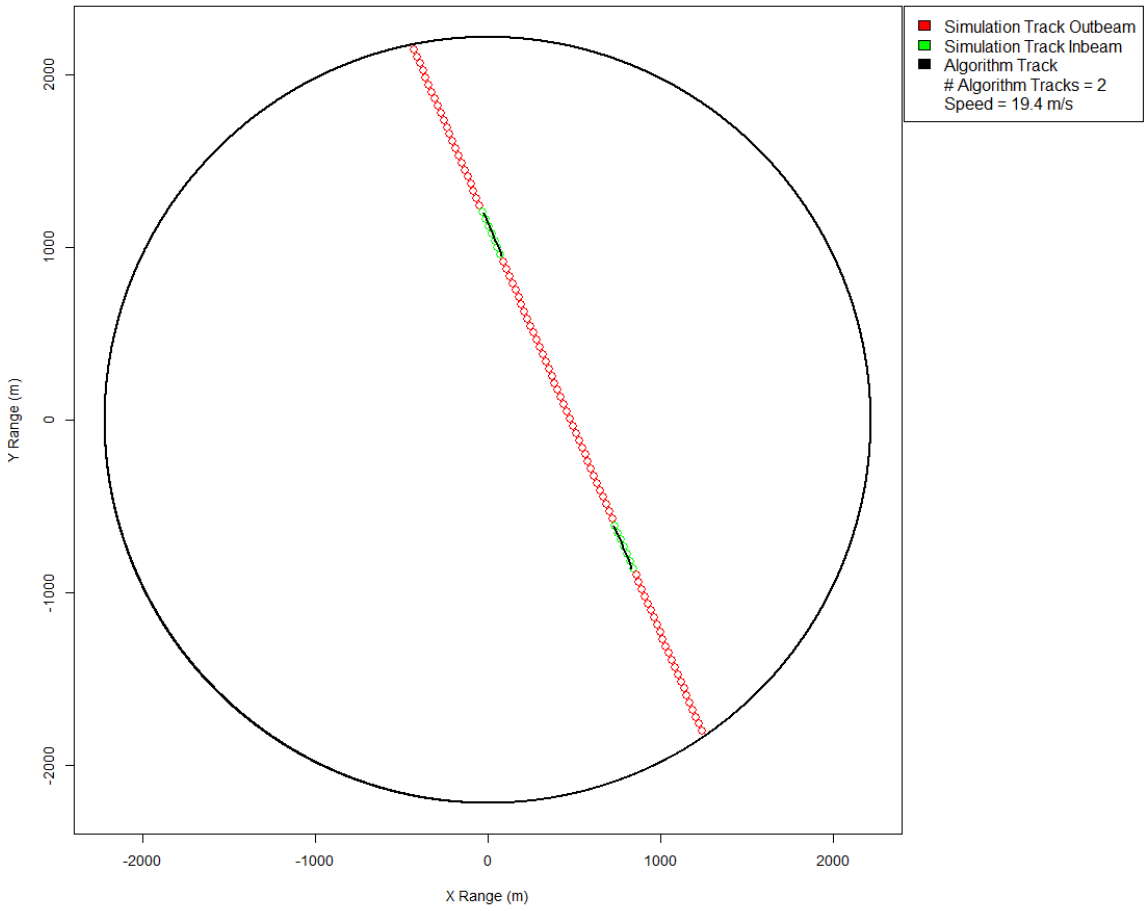

Figure 8: Example of a single simulated track crossing the radar beam twice and detected as separate tracks when applying the tracker algorithm and blip processing filter.

Table 5: Summary of the percentage of simulated targets or tracks detected in the radar beam after various levels of processing (described in Table 2).

Processing Level	Units	Original Numbers in Range of the Radar	Detected Numbers	Percentage Detected (%) in the Radar Beam
None	Targets	406542	37629	9.256
	Tracks	3480	1621	46.58
Blip	Targets	406542	32409	7.972
Tracker	Tracks	3480	1433	41.178

Table 6: Summary of the number and percentage of simulated tracks detected after applying the tracker algorithm and blip processing criteria. Misdetected refers to detected tracks that are composed of multiple different simulated tracks.

Detection	Number of Tracks Detected	Percentage Breakdown of Detected Tracks (%)
Detected Once	300	20.935
Detected Twice	1100	76.762
Detected Thrice	28	1.954
Detected Four Times	2	0.140
Misdetected	3	0.209
Total	**1433 (2610*)**	**100**

*number of tracks generated by the tracker algorithm that crossed within the radar range

After the simulated blipmovie was merged with a real raw archive from the actual study site, the number of targets and tracks detected in the radar beam increased considerably compared to the numbers detected from the simulated blipmovie without background (e.g. Figure 9). The number of simulated targets detected in the radar beam increased by

approximately 868% (blip processed) and 288% (blip and clutter processed), while the number of simulated tracks increased by roughly 573% after applying the tracker processing algorithm. These indicate over-detection of targets and tracks (Table 7).

Table 7: Summary of the percentage of simulated targets or tracks detected when merging with a real radar recording. Tracks were not adjusted for target speed.

Processing Level	Units	Original Numbers in the Radar Beam	Detected Numbers After Processing	Percentage After Processing (%)
Blip	Targets	37629	326580	867.894
Blip and Clutter	Targets	37629	108367	287.988
Tracker	Tracks	1621	9292	573.226

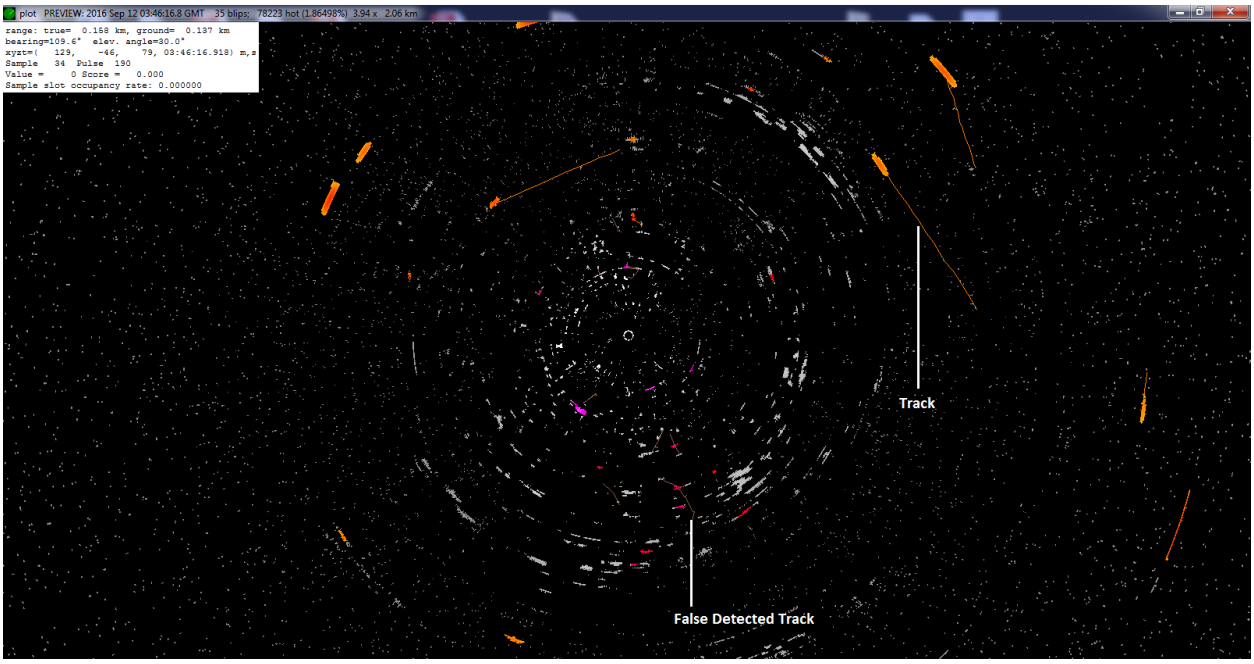

Figure 9: Example of a single scan showing simulated targets merged with real radar recording and indicating both correctly and incorrectly detected tracks (red tails). An individual tail shows the preceding position of the blips from the preceding scans that constitute a single track. Colored blips indicate targets, while greyed out blips indicate plausible clutter. Clutter may erroneously appear as targets when miscategorized.

The proportion of simulated targets varied considerably with altitude, regardless of processing level. The proportion of simulated targets detected in the radar beam increased somewhat rapidly with altitude and subsequently decreased abruptly when targets could no longer be detected in the beam (Figure 10). After applying the blip filtering criteria to increase realism, the proportion of simulated targets detected at lower altitudes were somewhat less when compared to the proportion of simulated targets detected without processing, up to the 500-600m bin. Beyond this altitude bin, the proportion of blip-processed simulated targets coincided well with the proportion of simulated targets detected without processing (~80%),

with few or none detected beyond 1100m. Overall, the proportion of simulated targets detected in the radar beam was less than 20% for every altitude bin, regardless of processing level.

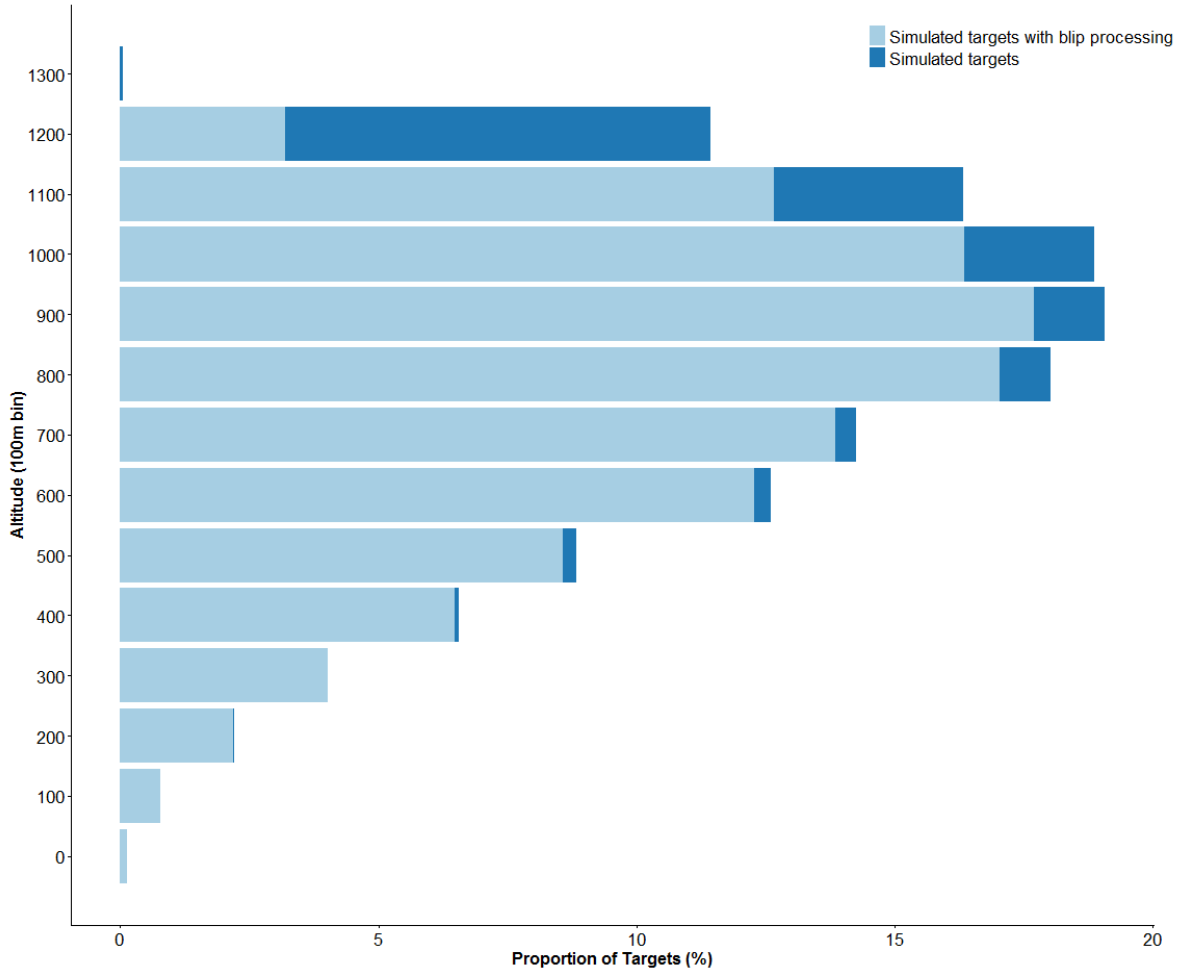

Figure 10: The proportion of simulated targets (n = 37629) and simulated targets filtered with blip processing (n = 32409) that were detected at least once in the radar beam out of all targets within range of the radar for each altitude band.

The proportion of simulated tracks detected at least once in the radar beam increased somewhat rapidly with altitude and subsequently decreased abruptly when tracks could no longer be detected in the beam (Figure 11). After applying the tracker algorithm and the blip

filtering criteria to increase realism, the proportion of simulated tracks detected at lower altitudes coincided closely with the proportion of simulated tracks detected without processing, although relatively few or none were detected at the highest and lowest altitude bins. Overall, the proportion of simulated tracks detected in the radar beam is small at lower altitudes, but increases incrementally towards 100% at higher altitudes. Moreover, the proportion of blip-and-tracker processed tracks detected in the radar beam is greater than 90% for every altitude bin, save for the highest and lowest altitude bins.

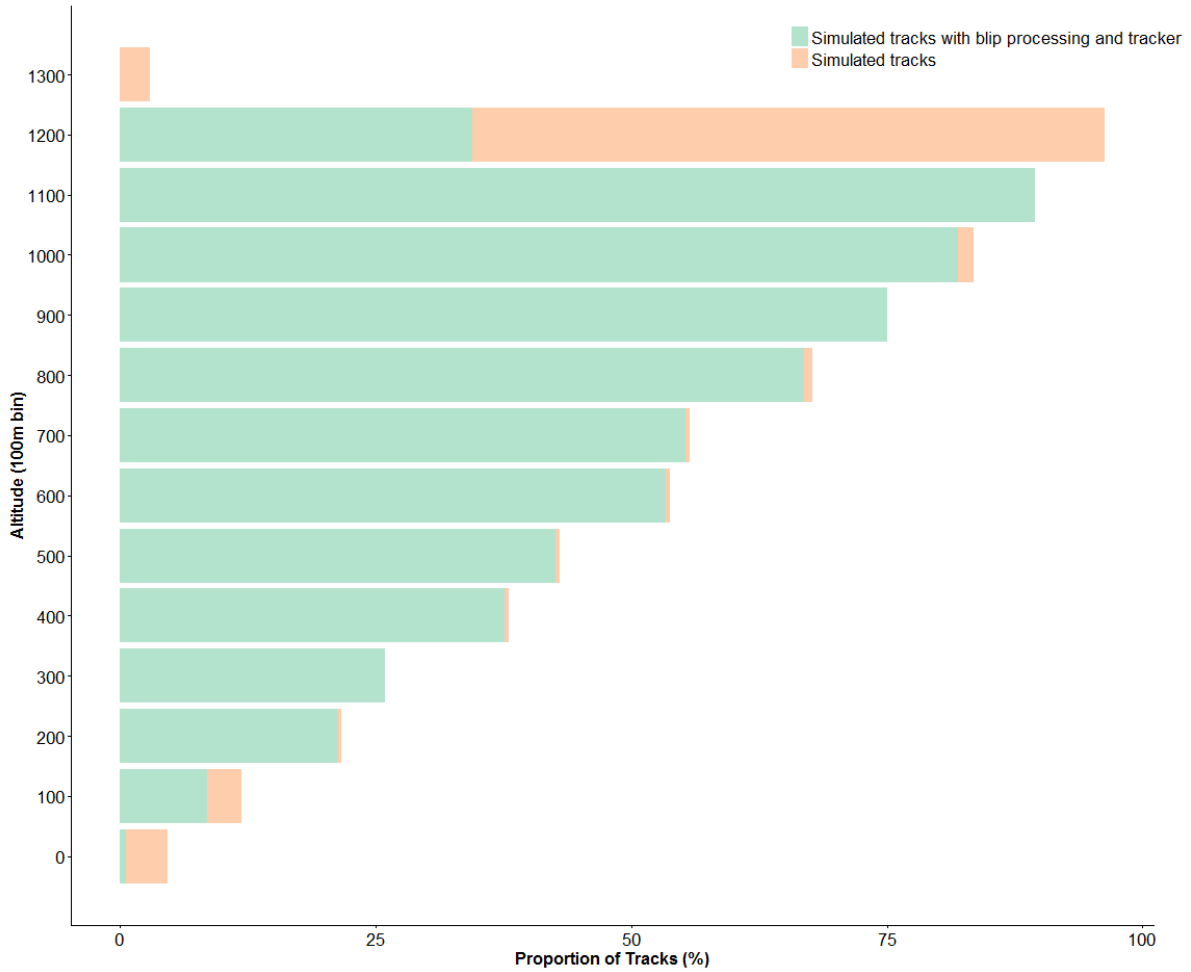

Figure 11: The proportion of simulated tracks within range of the radar (n = 3480) that were detected at least once in the radar beam (n = 1621) and that were recognized as tracks after applying the blip processing filter and tracker algorithm (n = 1433) in relation to altitude.

Applied Probabilities of Detection

When applying the POD from the simulated tracks that crossed the beam to a migratory bird dataset, the number of birds increased considerably in every altitude bin below 800 m, with the majority of that increase occurring below 400 m (Figure 12). The greatest proportional increase occurred in bin 0-100m. Few birds were detected above 800 m in the original bird dataset.

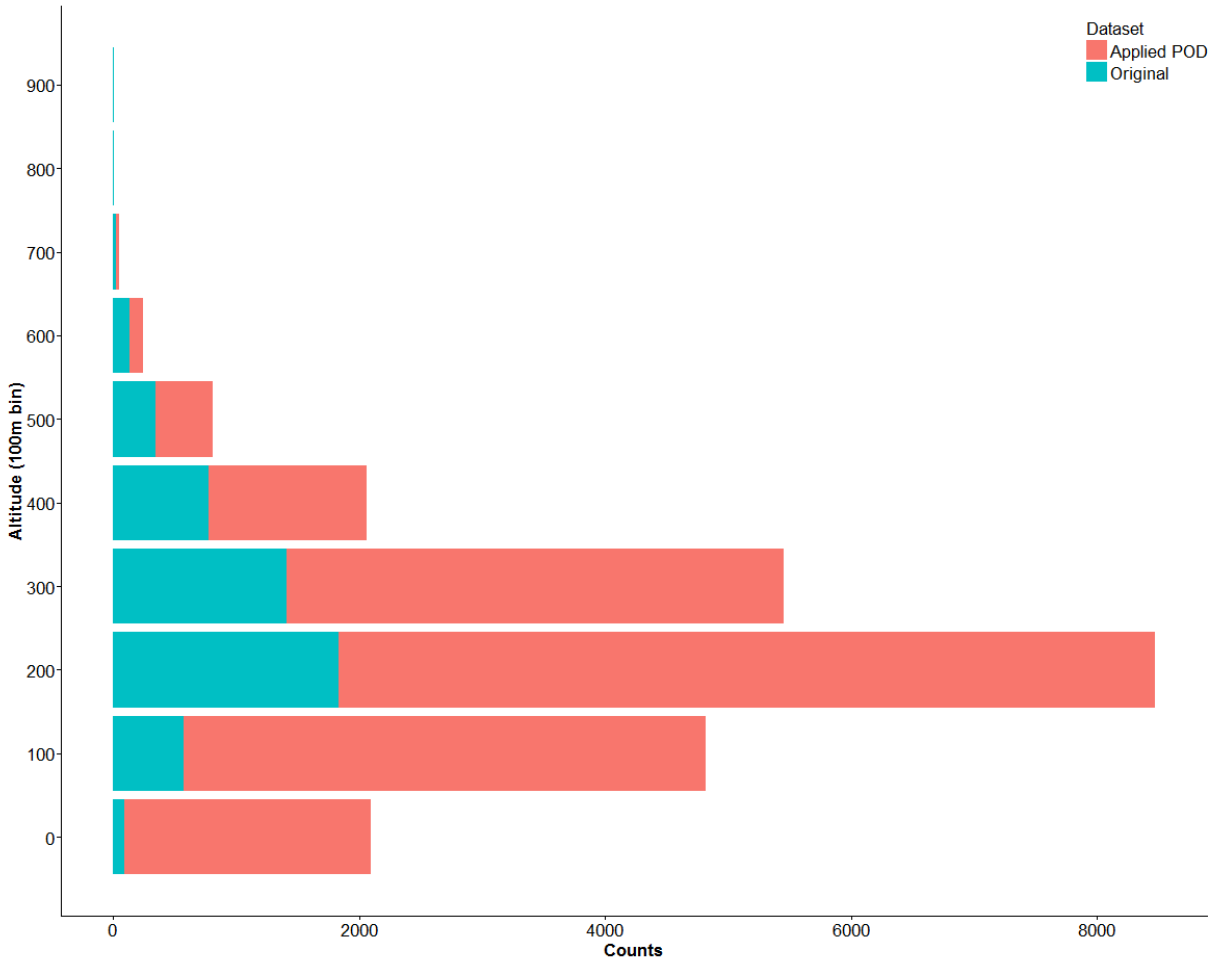

Figure 12: Estimated numbers of nocturnal birds in different altitude bands in a migratory dataset before and after adjusting for detection probabilities of the standalone simulation (see Figure 11).The original dataset was recorded between October 8th and 15th, 2013, in Essex County, Ontario.

When applying the POD from the simulated tracks that crossed the beam and were blip-and-tracker processed to a migratory bird dataset, the number of birds increased considerably in every altitude bin below 700 m, with the majority of that increase occurring below 400 m (Figure 13). The greatest proportional increase occurred in the lowest altitude bin (0-100 m), where the majority of clutter is likely detected.

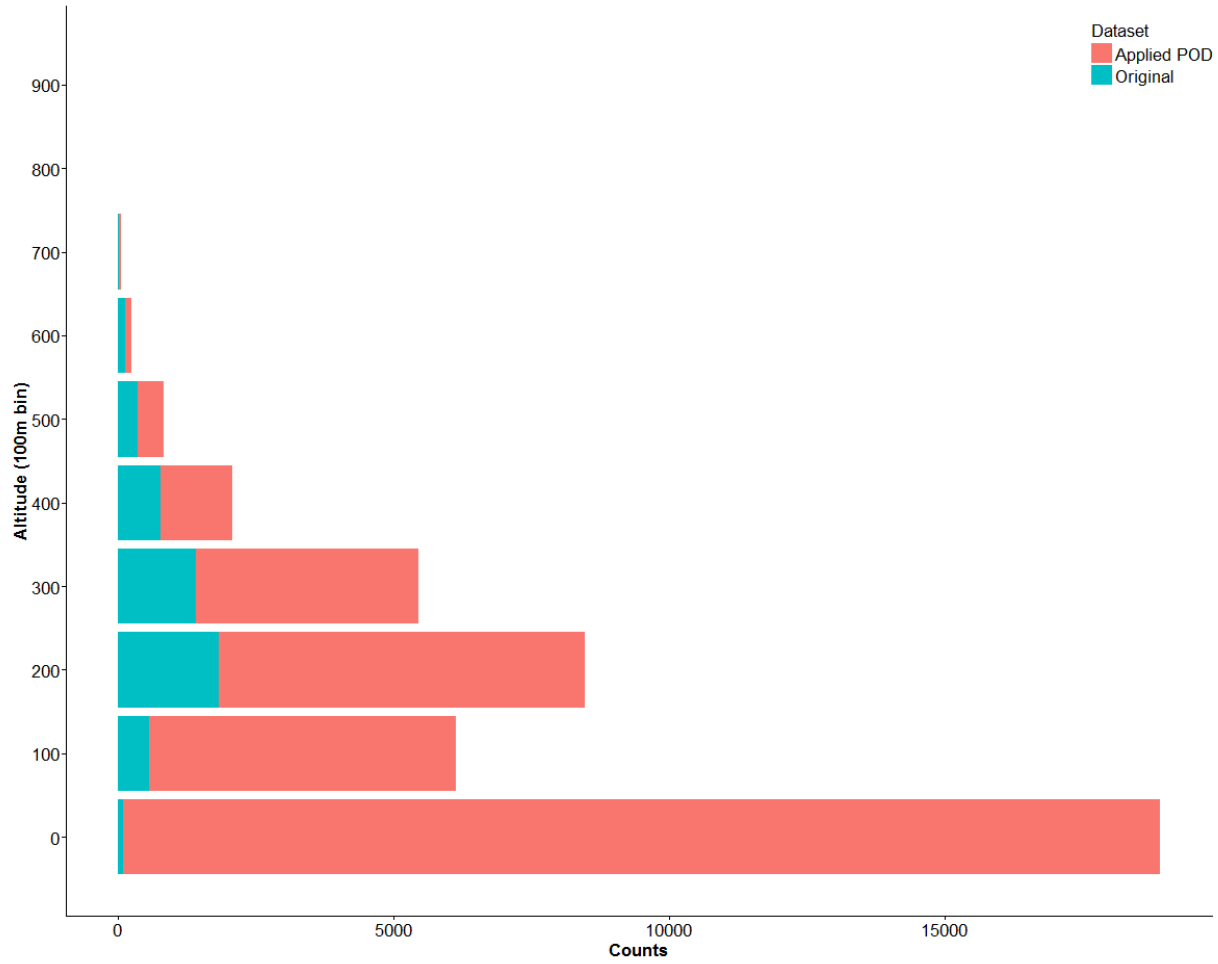

Figure 13: Estimated numbers of nocturnal birds in different altitude bands in a migratory dataset before and after adjusting for detection probabilities of the blip-and-tracker processed simulation (see Figure 11).The original dataset was recorded between October 8th and 15th, 2013, in Essex County, Ontario. The two highest altitude bins (800 m and 900 m) were too small to be shown graphically.

Discussion

I found considerable variation among radars units, specifically in relation to return echo strength with distance. Most studies rely on nominal power provided by manufacturers when estimating the number of birds flying past a given area (e.g. Mabee et al. 2006, Kahlert *et al.* 2012). Moreover, some studies set up multiple radars with identical nominal power to detect birds aloft (e.g. Burger, 2001). Therefore, it is imperative that bird radars are functioning at optimal performance prior to undertaking field measurements (Taylor *et al.* 2010).

My results indicate that the nominal beam width (+/- 3dB) is approximately 4° and the effective beam width is roughly 1.75 times larger than the nominal beam width. I found only two studies using bird radars that have previously attempted to quantify beam shape, with particular emphasis on effective beam width. Liechti *et al.* (1995) (using a radar in parallel with passive infrared) and Schmaljohann *et al.* (2008) (using the distribution of radar cross-sections at the centre of the beam) suggested an operational beam width ~2-2.5 times greater than nominal beam width. Some radar studies rely on the nominal beam width provided by manufacturers (e.g. Biebach *et al.* 2000) when estimating the numbers and distribution of birds (i.e. through the surveyed airspace volume of the radar beam).

The results from the simulation model indicate considerable variation in detectability with altitude. A smaller proportion of tracks were detected in lower altitude bands, suggesting many low flying birds are not detected by bird radars. Many published findings using marine radars indicate that most nocturnally migrating birds are detected at altitudes between 300-600 m (e.g. Mabee *et al.* 2006; Cooper & Ritchie, 1995; Gauthreaux 1972, 1991). Studies using more powerful radars (Mateos-Rodríguez & Liechti 2012; Schmaljohann *et al.* 2009) or weather

surveillance radars (LaSorte *et al.* 2015; Dokter *et al.* 2011) detect many more birds at higher altitudes than observed by marine radars. Thus it is important that bird radar studies obtain models of the detection probabilities with altitude prior to estimating the number of birds detected at different altitudes, as neglecting this might bias results towards higher flight altitudes (Dokter *et al.* 2013), or lower altitudes if many birds are above the radar.

My simulation relied on a standard set of processing algorithms that have been developed in radR (Taylor *et al.* 2010), and highlight a number of limitations in those algorithms. These limitations include the splitting of tracks and false detections when merging background clutter. For example, in the simulation the tracker algorithm frequently split tracks that crossed the beam twice. Also, removal of background clutter was often not successful as many tracks were erroneously created with non-targets, especially near the radar. It would be possible to refine or modify the detection and clutter suppression algorithms in radR and use the approach presented herein to test the effectiveness of the currently used algorithms or altogether different algorithms (e.g. machine learning algorithms Rosa *et al.* (2016) or proprietary algorithms). Moreover, different radar systems and/or antennas could also be used with the approach described here.

I suggest that all bird radar studies should include a component that estimates the probabilities of detection of targets flying past a radar. For example, Schmaljohann *et al.* (2008) and Dokter *et al.* (2013) attempted to empirically quantify detection probabilities with birds crossing the beam, but these methods have many limitations (e.g. human error, unable to estimate altitude of birds that are not detected, unknown radar cross-sections). My approach provides a viable alternative that could be applied in many situations. Many existing bird radars

(e.g. Detect: www.detect-inc.com; Accipiter: www.accipiterradar.com) use proprietary processing algorithms that have not been published, with no presented data on detectability. I suggest it is important to undertake similar exercises using approaches such as describe here to validate those or other algorithms used in bird detection to allow comparisons among sites and radars.

There were some challenges to overcome when analyzing data from the radars. Radar power differed among some of the radars, especially between Eagle (12.5 kW) and Plover (25 kW). However, given that magnetrons (microwave generator in each radar) in bird radars wear out at inconsistent rates over time, the radars likely differed in their power output level compared to factory levels, as was observed during field trials with Eagle being noticeably stronger. This is less of a problem in radars that use solid state technology (e.g. Detect radars), but other detectability issues raised in this study still apply. Moreover, the common power levels for marine radars used by zoologist range between 3 kW and 25 kW, which translates into an increase detection range of roughly 19% when doubling the peak power output (Taylor *et al.* 2010). For these reasons, power transmitted by the radars (P_t) remained unchanged. Also, exact information on whether the sphere was located at the centre of the beam could not be obtained. As a result, peak power per range bin was used as a proxy for the centre of the beam, which may have influenced the choice in extracted blips used to construct the power and beam shape models.

Future studies could improve on the simulation model to obtain more realistic results. One obvious way is to incorporate the actual radar cross-sections of different sized birds in flight. This could be achieved by recording a bird in flight in a laboratory setting (e.g. wind

tunnel) to obtain a model of the actual radar cross-section of a bird and how it changes over time in flight. Moreover, testing several detection and clutter suppression algorithms typically used in radar studies would help improve on the detection rates especially at lower altitudes, where conical scanning radars typically have narrow beam widths and clutter is abundant. Additionally, the simulation approach presented here could be repeated with data from other radars to show how detectability would be affected by power. An important step in future radar studies would be linking the estimated number of birds detected to species level. At present, radars cannot reliably estimate species aloft. However, the combination of radar findings and wildlife radio tracking systems (e.g. MOTUS, Francis *et al.* 2016) may offer promising ways to detect species composition and study behaviour patterns in large and geographically wide-ranging datasets.

LITERATURE CITED

Accipter. (2014). Retrieved August 29, 2016, from http://www.accipiterradar.com.

Biebach, H., Biebach, I., Friedrich, W., Heine, G., Partecke, J. & Schmidl, D. (2000). Strategies of passerine migration across the Mediterranean Sea and the Sahara desert: a radar study. *Ibis* **142:** 623-634.

Bruderer, B. (1997a). The study of bird migration by radar. Part 1: The technical basis. *Naturwissenschaften* **84:** 1-8.

Bruderer, B. (1997b). The study of bird migration by radar part 2: Major achievements. *Naturwissenschaften* **84:** 45-54.

Bruderer, B. (2003). The radar window to bird migration. In Berthold, P., Gwinner, E. & Sonnenschein, E. (eds) *Avian Migration*: 347–358. Heidelberg: Springer.

Burger, A. E. (2001). Using radar to estimate populations and assess habitat associations of Marbled Murrelets. *The Journal of Wildlife Management* **65:** 696-715.

Cabrera-Cruz S. A., Mabee T. J., & Villegas, R. V. (2013). Using theoretical flight speeds to discriminate birds from insects in radar studies. *Condor* **115:** 263–272.

Casement, M. B. (1966). Migration across the Mediterranean observed by radar. *Ibis* **108:** 461-491.

Cooper, B. A., & Ritchie, R. J. (1995). The altitude of bird migration in East-Central Alaska: a radar and visual study. *Journal of Field Ornithology* **66:** 590-608.

Day, R. H., Prichard, A. K., & Rose, J. R. (2005). Migration and Collision Avoidance of Eiders and Other Birds at Northstar Island, Alaska, 2001-2004: Final Report. ABR Inc., Environmental Research & Services, Fairbanks, AK. Prepared for BP Exploration (Alaska) Inc., Anchorage, AK, USA.

Deng, J., & Frederick, P. (2001). Nocturnal flight behavior of waterbirds in close proximity to a transmission powerline in the Florida Everglades. *Waterbirds: The International Journal of Waterbird Biology* **24:** 419-424.

Desholm, M., Gill, R., Bøvith, T., & Fox, A. D. (2014). Combining spatial modelling and radar to identify and protect avian migratory hot-spots. *Current Zoology* **60:** 680-691.

DeTect. (2016). Retrieved August 29, 2016, from http://www.detect-inc.com.

Dokter, A. M., Liechti, F., Stark, H., Delobbe, L., Tabary, P., & Holleman, I. (2011). Bird migration flight altitudes studied by a network of operational weather radars. *Journal of the Royal Society Interface* **8:** 30-43.

Dokter, A. M., Baptist, M. J., Ens, B. J., Krijgsveld, K. L., & van Loon, E. E. (2013). Bird radar validation in the field by time-referencing line-transect surveys. *PloS ONE* **8:** e74129. doi:10.1371/journal.pone.0074129.

Edwards, J., & Houghton, E. W. (1959). Radar echoing area polar diagram of birds. *Nature* **184:** 1059.

Francis, C. M., Taylor, P. D., & Crysler, Z. J. (2016). BOU Proceedings – Birds in time and space: avian tracking and remote sensing http://www.bou.org.uk/bouproc-net/tracking/poster-francis-etal.pdf.

Gauthreaux Jr, S. A. (1972). Behavioral responses of migrating birds to daylight and darkness: a radar and direct visual study. *The Wilson Bulletin* **84:** 136-148.

Gauthreaux Jr, S. A. (1991). The flight behavior of migrating birds in changing wind fields: radar and visual analyses. *American Zoologist* **31:** 187-204.

Gauthreaux Jr, S. A. (1984). The use of small mobile radars to detect, monitor, and quantify bird movements. In Harrison, M. J., Gauthreaux, S. A. & Abron-Robinson, L. A. (eds) Proceedings of the Wildlife Hazards to Aircraft Conference and Training Workshop: 121-131. Washington, DC: United States Department of Transport, Federal Aviation Administration.

Gauthreaux Jr, S. A., & Belser, C. G. (2003). Radar ornithology and biological conservation. *The Auk* **120:** 266-277.

Griffiths, H. D., Baker, C. J., Adamy, D. & Stimson, G. W. (2014). Stimson's Introduction to Airborne Radar. Edison, NJ: SciTech Publishing.

Harmata, A. R., Podruzny, K. M., Zelenak, J. R., & Morrison, M. L. (1999). Using marine surveillance radar to study bird movements and impact assessment. *Wildlife Society Bulletin* **27:** 44-52.

Kahlert, J., Leito, A., Laubek, B., Luigujõe, L., Kuresoo, A., Aaen, K., & Luud, A. (2012). Factors affecting the flight altitude of migrating waterbirds in Western Estonia. *Ornis Fennica* **89:** 241-253.

Lack, D., & Varley, G. C. (1945). Detection of birds by radar. *Nature* **156:** 446.

La Sorte, F. A., Hochachka, W. M., Farnsworth, A., Sheldon, D., Van Doren, B. M., Fink, D., & Kelling, S. (2015). Seasonal changes in the altitudinal distribution of nocturnally migrating birds during autumn migration. *Royal Society open science* **2:** 150347. http://dx.doi.org/10.1098/rsos.150347.

Liechti, F., Bruderer, B. & Paproth, H. (1995). Quantification of nocturnal bird migration by moonwatching: comparison with radar and infrared observations. *Journal of Field Ornithology* **66:** 457-468.

Mabee, T. J., Cooper, B. A., Plissner, J. H., & Young, D. P. (2006). Nocturnal bird migration over an Appalachian ridge at a proposed wind power project. *Wildlife Society Bulletin* **34:** 682-690.

Mateos-Rodríguez, M., & Liechti, F. (2011). How do diurnal long-distance migrants select flight altitude in relation to wind?. Behavioral Ecology **23**: 403-409.

R Core Team. 2015. R: A language and environment for statistical computing. R Foundation for Statistical Computing, Vienna, Austria. URL: http://www.R- project.org/.

Radar Navigation and Maneuvering Board Manual (7th ed.). (2001). Bethesda, MD: National Imagery and Mapping Agency.

Richards, M. A. (2005). Fundamentals of radar signal processing. New York, NY: Tata McGraw-Hill Education.

Rosa, D., Isabel, M., Marques, A.T., Palminha, G., Costa, H., Mascarenhas, M., Fonseca, C., & Bernardino, J. (2016). Classification success of six machine learning algorithms in radar ornithology. *Ibis* **158:** 28-42.

Schmaljohann, H., Liechti, F., Bächler, E., Steuri, T., & Bruderer, B. (2008). Quantification of bird migration by radar – a detection probability problem. *Ibis* **150:** 342-355.

Schmaljohann, H., Liechti, F., & Bruderer, B. (2009). Trans-Sahara migrants select flight altitudes to minimize energy costs rather than water loss. *Behavioral ecology and sociobiology* **63:** 1609-1619.

Shafique, K. & Shah, M. (2005). A noniterative greedy algorithm for multiframe point correspondence. *IEEE Transactions on Pattern Analysis and Machine Intelligence* **27:** 51-65.

Taylor, P. D., Brzustowski, J. M., Matkovich, C., Peckford, M. L., Wilson, D. (2010). radR: an open-source platform for acquiring and analysing data on biological targets observed by surveillance radar. *BMC Ecology* **10:** 22.

Thomas, P. J., Labrosse, A. K., Pomeroy, A. C., & Otter, K. A. (2011). Effects of weather on avian migration at proposed ridgeline wind energy sites. *The Journal of Wildlife Management* **75:** 805-815.

Vaughn, C. R. (1985). Birds and insects as radar targets: a review. *Proceedings of the IEEE* **73:** 205-227.

Villeneuve, T. J. (2015). Does the Lake Huron shoreline influence distributions, altitudes, and flight directions of nocturnally migrating birds? Master's thesis. Carleton University.

Williams, T. C. (1984). How to use marine radar for bird watching. *American Birds* **38:** 982-983.

Williams, T. C., Williams, J. M., Williams, P. G., & Stokstad, P. (2001). Bird migration through a mountain pass studied with high resolution radar, ceilometers, and census. *The Auk* **118:** 389-403.

APPENDIX A

Table A1: R-squared values and second-order polynomial models of the range bins that constitute the modelled beam shape. The last two range bins are modelled with the intercept only.

Range bin (m)	R^2	Intercept	β1	β2
522	0.9730	2720.7948	-0.2450	4.9319
527	0.8026	2955.9068	21.0169	-28.1756
532	0.9839	3071.8208	24.0134	-31.8184
536	0.9877	3081.8734	28.2692	-30.5134
541	0.9667	3036.3066	24.3212	-29.1844
546	0.8441	2912.9250	22.3104	-24.9294
550	0.4765	2742.0915	24.0449	-7.4067
555	NA	2703.4550	0	NA
560	NA	2697.7200	0	NA

APPENDIX B:

An R script of the genblips simulation and an R script of the associated configuration parameters as implemented into radR

Genblips Simulation Script

```
##  svn $Id: genblips.plugin.R 692 2010-12-07 15:46:22Z john $
##
###< Copyright
##  radR : an R-based platform for acquisition and analysis of radar data
##  Copyright (C) 2006-2009 John Brzustowski
##
##  This program is free software; you can redistribute it and/or modify
##  it under the terms of the GNU General Public License as published by
##  the Free Software Foundation; either version 2 of the License, or
##  (at your option) any later version.
##
##  This program is distributed in the hope that it will be useful,
##  but WITHOUT ANY WARRANTY; without even the implied warranty of
##  MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
##  GNU General Public License for more details.
##
##  You should have received a copy of the GNU General Public License
##  along with this program.  If not, see <http://www.gnu.org/licenses/>.
###>

##   genblips.plugin.R
##
## A simple plugin for generating artificial blips from a dynamic
## model of targets.  Provides a reader port.
##


## functions to be loaded
load("C:/users/chris/desktop/sim4/paint_blips_polynomial_sim4.rda") ## df2
polynomial models
load("C:/users/chris/desktop/sim4/interval_sim4.rda") ## df6 sample
intervals
load("C:/users/chris/desktop/sim4/usrp_eagle_R251.rda") ## URSP conversion
function
load("C:/users/chris/desktop/sim4/powerfunction_fourthpower_bindata_Plover
Eagle.rda")  # power function

## converts mWatt/cm2 into dBm
dBm <- function(mW_cm2){
  10*log10(mW_cm2/1e4)
}
```

```
MYCLASS="genblips"

about = function() {
  return("This plugin provides a reader port that provides random
artificial blips\naccording to a dynamic model.\nSome of the model
parameters are actually R expressions,\nwhich you can modify.\n\nTargets
within the radar beam are classified as blips, whereas\n those outside the
beam are classified as cold.")
}

get.ports = function() {

  rv <- list()

  make.port <- function(name, id, is.source, is.sink) {
    structure(strictenv(
      ## common to all ports
      name = name,
      id = id,
      is.source = is.source,
      is.sink = is.sink,
      is.live = FALSE,
      is.file = FALSE,
      is.seekable = FALSE,
      can.specify.start.time = FALSE,
      config = list(),
      file.ext = "",
      has.toc = FALSE
    ),
    class = c(MYCLASS, "strictenv"))
  }

  rv <- list()

  ##                     name    id  source  sink
  rv[[1]] <- make.port("Artificial blip generator", 1,  TRUE, FALSE)
  rv
}

load = function() {
  ## initialize state variables here
  rss.dyn.load("genblips", in.dir.of=plugin.file)
}

unload = function(save.config) {
  rss.dyn.unload("genblips")
}


get.menus = function() {
  list(
    sources = list(
      titles = "Artificial blip generator",
      menu = list(
```

```
        options = "no-tearoff",
        "Start" = function() rss.set.port(get.ports()[[1]])
      )
    ),
    plugin = list (
      list(option="choose.any",
           on.set = function(n, v) {
             if (n == 1)
               generate.csv.file <<- v
             else if (n == 2)
               only.export.visible <<- v
           },
           "Generate a truetracks.csv file" = generate.csv.file,
           "Only export visible targets to .CSV file" =
only.export.visible
      ),
      radius=list ("gauge",
                   label = "radius of plotted targets and blips",
                   range = c(0, 100),
                   increment = 1,
                   value = radius,
                   on.set = function(x) { radius <<- x;
reset.blips.deltas()}
      ),
      list ("gauge",
            label = "maximum number of simultaneous targets:
target.n.max",
            range = c(0, 1000),
            increment = 1,
            value = target.n.max,
            on.set = function(x) { target.n.max <<- x}
      ),
      list ("gauge",
            label = "mean new targets per scan:   target.new.mu"
            ,
            range = c(0, 1000),
            increment = 1,
            value = target.new.mu,
            on.set = function(x) { target.new.mu <<- x}
      ),
      list ("gauge",
            label = "mean noise blips per scan:   noise.blips.mu",
            range = c(0, 1000),
            increment = 1,
            value = noise.blips.mu,
            on.set = function(x) { noise.blips.mu <<- x}
      ),
      list ("gauge",
            label = "mean target radar cross section; (UNUSED):
target.rcs.mu",
            range = c(0, 1000),
            increment = 1,
            value = target.rcs.mu,
            on.set = function(x) { target.rcs.mu <<- x}
```

```
      ),
      list ("gauge",
            label = "mean noise radar cross section; (UNUSED):
noise.rcs.mu",
            range = c(0, 1000),
            increment = 1,
            value = noise.rcs.mu,
            on.set = function(x) { noise.rcs.mu <<- x}
      ),
      list ("gauge",
            label = "multiplies accelerations each scan:
target.accel.decay",
            range = c(0, 5),
            increment = .01,
            value = target.accel.decay,
            on.set = function(x) { target.accel.decay <<- x}
      ),
      list ("gauge",
            label = "maximum target speed in km/hr:   target.max.speed",
            range = c(0, 1000),
            increment = 1,
            value = target.max.speed,
            on.set = function(x) { target.max.speed <<- x}
      ),
      list ("string",
            label = "number of new targets in scan #s:   target.new.num",
            width = 40,
            height = 2,
            value = as.character(target.num.new),
            val.check = function(x) tryCatch({parse(text=x); TRUE},
error=function(e)FALSE),
            on.set = function(x) { target.num.new <<- parse(text=x)}
      ),
      list ("string",
            label = "n x 3 matrix of initial target locations:
target.x0",
            width = 40,
            height = 2,
            value = as.character(target.x0),
            val.check = function(x) tryCatch({parse(text=x); TRUE},
error=function(e)FALSE),
            on.set = function(x) { target.x0 <<- parse(text=x)}
      ),
      list ("string",
            label = "n x 3 matrix of initial target velocities:
target.v0",
            width = 40,
            height = 2,
            value = as.character(target.v0),
            val.check = function(x) tryCatch({parse(text=x); TRUE},
error=function(e)FALSE),
            on.set = function(x) { target.v0 <<- parse(text=x)}
      ),
      list ("string",
```

```
            label = "n x 3 matrix of initial target accelerations:
target.a0",
            width = 40,
            height = 2,
            value = as.character(target.a0),
            val.check = function(x) tryCatch({parse(text=x); TRUE},
error=function(e)FALSE),
            on.set = function(x) { target.a0 <<- parse(text=x)}
      ),
      list ("string",
            label = "vector of n target radar cross sections:
target.rcs",
            width = 40,
            height = 2,
            value = as.character(target.rcs),
            val.check = function(x) tryCatch({parse(text=x); TRUE},
error=function(e)FALSE),
            on.set = function(x) { target.rcs <<- parse(text=x)}
      ),
      list ("string",
            label = "number of noise blips in scan #s:   noise.num.blips",
            width = 40,
            height = 2,
            value = as.character(noise.num.blips),
            val.check = function(x) tryCatch({parse(text=x); TRUE},
error=function(e)FALSE),
            on.set = function(x) { noise.num.blips <<- parse(text=x)}
      ),
      list ("string",
            label = "n x 3 matrix of noise blip locations:   noise.x0",
            width = 40,
            height = 2,
            value = as.character(noise.x0),
            val.check = function(x) tryCatch({parse(text=x); TRUE},
error=function(e)FALSE),
            on.set = function(x) { noise.x0 <<- parse(text=x)}
      ),
      list ("string",
            label = "vector of n noise blip radar cross sections:
noise.rcs",
            width = 40,
            height = 2,
            value = as.character(noise.rcs),
            val.check = function(x) tryCatch({parse(text=x); TRUE},
error=function(e)FALSE),
            on.set = function(x) { noise.rcs <<- parse(text=x)}
      ),
      list ("string",
            label = "number of noise blips in scan s:   noise.num.blips",
            width = 40,
            height = 2,
            value = as.character(noise.num.blips),
            val.check = function(x) tryCatch({parse(text=x); TRUE},
error=function(e)FALSE),
```

```
            on.set = function(x) { noise.num.blips <<- parse(text=x)}
      )
    )
  )
}

reset.blip.deltas <- function() {
  blip.deltas <<- outer(-radius:radius, scan.info$samples.per.pulse * (-
39:39), `+`)
}


paint.blips <- function(scan.mat, class.mat, xrad, yrad, rcs,
inbeam,lk,inbeam2) {
  ## given "targets" at radar coordinates (xrad, yrad)
  ## and with radar cross section rcs
  ## paint plausible representations of the
  ## targets into the scan and class matrices
  ## (which must both be extmats already dimensioned
  ## to match scan.info[c("pulses", "samples.per.pulse")]
  ## convert xy locations to sample, pulse pairs
  ##
  ## where x is metres East of the radar, and y is metres North
  ##
  ## Returns an n x 2 integer matrix with columns sample and pulse;see
rss.xy.to.sp() and the console or radR
  sp <- rss.xy.to.sp(xrad, yrad)

  ## keep targets in the beam
  keep <- inbeam2

  ## stucture remains the same as sp
  sp <- sp[keep, ,drop = FALSE]
  if (!length(sp))
    return()

  ## get linear sample-pulse coordinates for centre of each blip
  ## will produce a vector with considerably large numbers
  ## scan.info is taken from genblips.conf
  lsp <- as.integer((sp[, 2] - 1) * (scan.info$samples.per.pulse) + sp[,
1])

  ## compute linear coordinates of all samples in all blips and wrap
around at 0-degree cut
  ## gets the dimensions of the extmat
  ## so nn is typically 1024 * 4096 = 4194304, though this changes with
setup values in genblips.conf
  nn <- prod(dim(scan.mat))
  lcc <- 1 + (outer(blip.deltas, lsp, `+`) - 1) %% nn

  ## paint in scan and class matrices;

  ## builds the blip using polynomial models and sample intervals
```

```
  rangef <- matrix(NA,dim(blip.deltas)[1],dim(blip.deltas)[2], byrow =
TRUE)
  for (i in seq(-radius:radius)) {
    for (j in seq(-39,39)) {
      if(is.na(df2[[i]]$coefficient[3])) {
        rangef[i,j+length(seq(-39,39))/2 + length(seq(-39,39))%%2/2] <-
df2[[i]]$coefficient[2]*
          (as.numeric(unlist(df6[[i]])) + j*as.numeric(unlist(df6[[i]])))
+ df2[[i]]$coefficient[1]
        } else {
          rangef[i,j+length(seq(-39,39))/2 + length(seq(-39,39))%%2/2] <-
df2[[i]]$coefficient[3]*
            (as.numeric(unlist(df6[[i]])) +
j*as.numeric(unlist(df6[[i]])))^2 +
          df2[[i]]$coefficient[2]*(as.numeric(unlist(df6[[i]])) +
j*as.numeric(unlist(df6[[i]]))) +
          df2[[i]]$coefficient[1]
        }
    }
    }
  rangef <- as.integer(rangef)

  test <- function(x) {
    ## re-scales lk (int2 or max power) onto rangef
    ## remove pixels that fall below a threshold
    uin <- as.integer(rangef - (max(rangef) - rep(x, each =
length(blip.deltas))))
    if(any(uin < 2250)){
      t <- which(uin < 2250)
      uin[t] <- 0L
      }else{
        uin
        }
    return(uin)
  }

  ## if blips are in beam paint them into the radR scan matrix
  scan.mat[lcc]  <- test(lk[keep])

  ## classify pixels as excluded if below threshold
  blip.class <- function(){
    if(any(as.logical(test(lk[keep])) == !FALSE)){
      bc <- as.integer(as.logical(test(lk[keep])) * 1)
      bc[which(bc == 0L)] <- as.integer(RSS$CLASS.VAL$excluded)
      bc <- as.integer(bc)
      return(bc)
      }else{
        bc <- rep(RSS$CLASS.VAL$excluded, each=length(blip.deltas))
      }
    }

  ## paint blip classfication into class matrix
  class.mat[lcc] <- as.integer(blip.class())
  }
```

```
globals = list (
  as.character.genblips = function(x, ...) {
    sprintf("radR interface port: %s",
            MYCLASS
    )
  },

  print.genblips = function(x, ...) {
    ## print a description of this port
    cat (as.character(x) %:% "\n")
  },

  config.genblips = function(port, ...) {
    ## send or get configuration options from a port
    ## for example, this is how to set the filename for a port:
    ## config(port, filename="whatever.dat")
    ## if opts == NULL, the current configuration is returned.
    ## if opts != NULL, the configuration is set
    ## and TRUE is returned on success, FALSE on error

    opts <- list(...)
    if (length(opts) != 0) {
      for (opt in names(opts)) {
        switch(opt,
               filename = {
                 port$config$filename <- opts[[opt]]
                 port$file.basename <- strsplit(opts$filename, "\\.[a-zA-
Z]*$")[[1]]
               },
               {
                 rss.plugin.error("raw: unknown configuration option for
port: " %:% opt)
                 return(NULL)
               }
        )
      }
    }
    return(port$config)
  },

  ##   get.contents.genblips = function(port, ...) {
  ##     ## gets the contents of the current (filetype)
  ##     ## source, namely
  ##     ## a list with three elements
  ##     ## num.images: number of images in each run
  ##     ## start.time:  starting timestamp of each run
  ##     ## end.time:    ending timestamp of each run

  ##     port$contents <- port$bl[[1]]
  ##     class(port$contents$start.time) <- "POSIXct"
  ##     class(port$contents$end.time) <- "POSIXct"
  ##     port$first.scan <- cumsum(c(1, port$contents$num.scans))
  ##     return(port$contents)
```

```
  ##   },

  end.of.data.genblips = function(port, ...) {
    ## return TRUE if there is no data left to be read
    ## on this port (e.g. if the end of a tape run has been hit)
    return(scan.num >= num.scans)
  },

  get.scan.info.genblips = function(port, ...) {
    ## gets the header information for the next scan
    ## along with probably caching this information internally,
    ## this returns a list with these elements:
    ##
    ## pulses: number of pulses in this scan
    ##
    ## samples.per.pulse: number of samples per pulse
    ##
    ## bits.per.sample: number of bits per sample
    ##
    ## timestamp: POSIXct-style timestamp of start of scan, including
fractional seconds
    ##
    ## duration: length of this scan (milliseconds) (i.e. rotation time
    ## for radar)
    ##
    ## sample.dist: what distance increment each sample represents
(metres)
    ##
    ## first.sample.dist: distance at which first sample begins (metres)
    ##
    ## bearing:  how many degrees clockwise from north is the first
    ##                 pulse?
    ##
    ## orientation: rotation direction: +1 = clockwise, -1 =
counterclockwise

    ## we simply copy these parameters from the plugin's config,
    ## which is in scope

    scan.num <<- 1 + scan.num
    si <- scan.info
    si$timestamp <- init.time + scan.num * si$duration / 1000
    return(si)
  },

  get.scan.data.genblips = function(port, extmat, ...) {

    ## empty the extmats corresponding to raw data and class
    dim(extmat) <- c(scan.info$samples.per.pulse, scan.info$pulses)
    zero(extmat)
    dim(RSS$class.mat) <- dim(extmat)
    zero(RSS$class.mat)
```

```
    ## drop any targets which are no longer in range or which have hit the
ground
    targets <<- subset(targets, range <= 1.6 * max.range & z > 0)

    ## move any remaining targets
    if (nrow(targets)) {
      ## determine the time to next encounter for each target
      .C("next_encounter", nrow(targets), 1L, targets$x, targets$y,
targets$z, targets$vx, targets$vy, targets$vz, targets$ax, targets$ay,
targets$az, targets$timestamp, 1000 / scan.info$duration, DUP=FALSE)
    }

    ## add more blips

    if (nrow(targets) < target.n.max) {
      n <- min(target.n.max - nrow(targets), eval(target.num.new))
      if (n) {
        new <- nrow(targets)+(1:n)
        targets[new ,] <<- cbind(rep(0, n),
## scan.no
                                 (1:n) + max.target.id,
## id
                                 rep(0, n),
## age
                                 rep(0, n),
## timestamp
                                 eval(target.x0),
## x, y, z
                                 eval(target.v0),
## vx, vy, vz
                                 eval(target.a0),
## ax, ay, az
                                 rep(0, n), rep(0, n), rep(0, n), rep(0,
n), rep(0, n), ## range, xyrange, azi, speed, elev
                                 eval(target.rcs), rep(FALSE, n),
rep(FALSE, n),        ## rcs, visible, visible2
                                 double(n), double(n), double(n),
double(n), double(n), ## x.rad, y.rad, z.rad, int, int2
                                 rep(0, n), rep(0, n))
## area, area2


        max.target.id <<- max.target.id + n
        ## timestamp corresponding to time at top of scan plus offset in
sweep
        targets$timestamp[new] <<- init.time + scan.num *
scan.info$duration/1000 * (pi / 2 - atan2(targets$y[new], targets$x[new]))
/ (2 * pi)
       }
    }

    ## update age, range, speed, elevation angle and visibility

    if (nrow(targets)) {
```

```
      targets$age <<- targets$age + 1
      targets$range <<- with(targets, sqrt(x^2+y^2+z^2))
      targets$xyrange <<- with(targets, sqrt(x^2+y^2))
      targets$speed <<- with(targets, sqrt(vx^2+vy^2+vz^2))
      targets$scan.no <<- scan.num

      too.fast <- targets$speed > target.max.speed
      too.fast <- ifelse(is.na(too.fast), FALSE, too.fast)
      if (sum(too.fast) > 0) {
        ## for speeds exceeding the maximum, scale the velocity so speed
equals the maximum
        adj <- target.max.speed / targets$speed[too.fast]
        targets$vx[too.fast] <<- targets$vx[too.fast] * adj
        targets$vy[too.fast] <<- targets$vy[too.fast] * adj
        targets$vz[too.fast] <<- targets$vz[too.fast] * adj
      }
      targets$elev <<- with(targets, deg(asin (z / range)))
      ## power function
      targets$int   <<- with(targets,
eaglef0UsrpValMap(dBm(exp(log(52186587304) -
(unname(coef(nl.fit_four_m_PE))*log(range))))))
      targets$int   <<- with(targets, ifelse(is.nan(int) | is.na(int)
|is.infinite(int) | int < 0 | is.null(int), 4095, int))
      ## peak range bin function from polynomial models
      intf          <- function(x) {df2[[4]]$coefficient[3]*x^2 +
df2[[4]]$coefficient[2]*x + df2[[4]]$coefficient[1]}
      ## power of target from off beam centre
      targets$int2  <<- with(targets,
                             ifelse(
                               abs(elev - RSS$scan.info$antenna.angle) <=
RSS$scan.info$antenna.aperture.v / 2
                               & with(targets, abs(elev -
RSS$scan.info$antenna.angle) <=
                                                 RSS$scan.info$antenna.aperture.v /
2 & int >= 2250 & int <= 4095) == TRUE,
                               targets$int - abs(as.numeric(intf(-
(as.numeric(df2[[4]]$coefficient[2]))/(2*(as.numeric(df2[[4]]$coefficient[
3])))))) -
                                                           intf(elev -
RSS$scan.info$antenna.angle)),targets$int))
      ## is the target visible
      targets$visible <<- with(targets, abs(elev -
RSS$scan.info$antenna.angle) <=
                                 RSS$scan.info$antenna.aperture.v / 2 &
int2 >= 2250 &
                                 int2 <= 4095 & range < 2561.165 & xyrange
< 2218.033953)
      targets$visible2 <<- with(targets, visible)
      ##unused
      targets$area <<- with(targets,
                            #if(!length(RSS$patches$area[])){
                            # 0
                            #} else{
                            if(any(visible2 == TRUE)){
```

```
                              vi <- which(visible2 == TRUE)
                              #for(i in seq_along(vi)){visible2[vi[i]] <-
abs(RSS$patches$area[i])}
                              for(i in seq_along(vi)){visible2[vi[i]] <-
i}
                              visible2
                              #  #visible2[which(visible2)] <-
sapply(seq_along(RSS$patches$area[]),function(x) RSS$patches$area[x])
                              # visible2[which(any(visible2 == TRUE))]
                            }else{
                              NA
                            }
                            #}

      )

      #unused
      targets$area2 <<- with(targets, RSS$num.blips)
      targets$azi <<- with(targets, atan2(y, x))
      cr <- cos(rad(RSS$scan.info$antenna.angle))
      targets$x.rad <<- with(targets, range * cr * cos(targets$azi))
      targets$y.rad <<- with(targets, range * cr * sin(targets$azi))
      targets$z.rad <<- with(targets, range *
sin(rad(RSS$scan.info$antenna.angle)))

    }

    ## generate noise blips
    n <- eval(noise.num.blips)
    if (n) {
      noise.x <- eval(noise.x0)
      ## convert noise locations to what would be seen by the radar
      noise.range <- sqrt(apply(noise.x^2, 1, sum))
      noise.azi <- atan2(noise.x[,2], noise.x[,1])
      cr <- cos(rad(RSS$scan.info$antenna.angle))
      noise.xrad <- noise.range * cr * cos(noise.azi)
      noise.yrad <- noise.range * cr * sin(noise.azi)
      noise.rcs <- eval(noise.rcs)
    } else {
      noise.xrad <- NULL
      noise.yrad <- NULL
      noise.rcs <- NULL
    }

    ## paint blips into scan and class matrices
    if (nrow(targets) + length(noise.xrad) > 0) {
      paint.blips(extmat, RSS$class.mat, c(targets$x, noise.xrad),
c(targets$y, noise.yrad), c(targets$rcs, noise.rcs), c(targets$visible,
rep(TRUE, length(noise.xrad))), c(targets$int2, NULL), c(targets$visible2,
NULL))
      RSS$num.blips <- sum(targets$visible, length(noise.xrad))
    } else {
      RSS$num.blips <- 0
      ### RSS$num.hot.samples  <- nrow(targets)
```

```
    }
    RSS$num.hot.samples <- RSS$num.blips * (2*radius+1)^2
    ### RSS$num.hot.samples  <- length(which(!(targets$visible)))

    RSS$new.have.valid$stats <- FALSE
    RSS$new.have.valid$scores <- FALSE
    RSS$new.have.valid$classification <- TRUE

    return(extmat)
  },

  start.up.genblips = function(port, ...) {
    ## disable the blip finding controls
    rss.gui(ENABLE_CONTROLS, "blip.finding", FALSE)
    scan.num <<- 0
    #num.blips <<- RSS$num.blips
    init.data()
    RSS$skip[c("calculate.scores", "classify.samples", "update.stats")] <-
c(TRUE, TRUE, TRUE)
    #RSS$scans.to.learn <- 0
    RSS$scans.to.learn <<- 0
    rss.enable.hook("ONPLAY", name)
    rss.enable.hook("ONSTOP", name)
    TRUE
  },

  shut.down.genblips = function(port, ...) {
    ## re-enable the blipping controls
    rss.gui(ENABLE_CONTROLS, "blip.finding", TRUE)
    rss.disable.hook("ONPLAY", name)
    rss.disable.hook("ONSTOP", name)
    RSS$skip[c("calculate.scores", "classify.samples", "update.stats")] <-
c(FALSE, FALSE, FALSE)
  },

  new.play.state.genblips = function(port, new.state, old.state, ...) {
    ## indicate to this port that radR is
    ## changing play state.

  }
)  ## end of globals

degg <- function(rad) {(rad * 180) / (pi)}
radd <- function(deg){(deg/180) * pi}

init.data <- function() {
  ## max range
  max.range <<- (1468.954671/sin(radd(RSS$scan.info$antenna.angle +
3.515625)))
  init.time <<- as.numeric(Sys.time())
  ## add atomic classes of targets
  targets <<- data.frame(scan.no=integer(),id=integer(), age=integer(),
timestamp=double(),
                         x=double(), y=double(), z=double(),
```

```
                          vx=double(), vy=double(), vz=double(),
                          ax=double(), ay=double(), az=double(),
                          range=double(), azi=double(), speed=double(),
elev=double(),
                          rcs=double(), visible=logical(),
visible2=logical(),
                          x.rad=double(), y.rad=double(), z.rad=double(),
int=double(), int2=double(),xyrange=double()
                          ,area=integer(), area2=double())
  max.target.id <<- 0
  reset.blip.deltas()
}

hooks = list(

  SCAN_INFO = list (enabled = TRUE, read.only = FALSE,
                    f = function(si) {
                      ## assign to the global scan.info object,
                      ## because get.scan.data.genblips needs an existing
RSS$scan.info
                      RSS$scan.info <- si
                    }),

  ONPLAY = list( enabled = FALSE, read.only = FALSE,
                 f = function(...) {
                   RSS$scans.to.learn <- 0
                 }),

  ONSTOP = list( enabled = FALSE, read.only = FALSE,
                 f = function(...) {
                   scan.num <<- 0
                   #num.blips <<- RSS$num.blips
                   zero(RSS$scan.mat)
                   zero(RSS$class.mat)
                   #zero(RSS$patches)
                   init.data()
                   if (!is.null(csv.file)) {
                     close(csv.file)
                     close(csv.file2)
                     csv.file <<- NULL
                     csv.file2 <<- NULL
                   }
                 }),
  ## write out data
  DONE_SCAN = list (enabled = TRUE, read.only = TRUE,
                    f = function(...) {
                      if (RSS$previewing ||
!RSS$have.valid$classification)
                        return()
                      if (generate.csv.file) {
                        if (is.null(csv.file)) {
                          csv.file <<- file(csv.filename, "w")
                          csv.file2 <<-file("patch_stats.csv","w")
```

```
cat(c("scan.no,track.no,blip.no,timestamp,x,y,z,vx,vy,vz,ax,ay,az,range,az
i,speed,elev.ang,rcs,visible,visible2,xrad,yrad,zrad,int,int2,xyrange,area
,area2"), "\n", file=csv.file)

cat(c("patch.no,scan.no,x,y,z,t,ns,area,int,max,aspan,rspan,perim,range_an
tenna.angle"), "\n", file=csv.file2)
                        }
                        if (only.export.visible) {
                          if (sum(targets$visible))
                            write.table(subset(targets, visible),
col.names=FALSE, row.names=FALSE, file=csv.file, append=TRUE, sep=",")
                        } else {
                          if (nrow(targets))
                            write.table(targets, col.names=FALSE,
row.names=FALSE, file=csv.file, append=TRUE, sep=",")
                          ## so use options (warn =-1)in radR console to
eliminate warnings
                          ## FIX ME

write.table(cbind(RSS$current.scan,suppressWarnings(do.call(cbind,lapply(u
nname(RSS$patches), matrix, ncol = 1, byrow = TRUE)))),
                                      col.names=FALSE, row.names=T,
file=csv.file2, append=TRUE, sep=",", na = "NA")

                        }
                      }
                    })

) ## end of hooks list

## additional state variables for this plugin

csv.file = NULL  ## connection for CSV output
max.range = 0
scan.num = 0  ## index of current scan
init.time = NULL ## time at which this run started
max.target.id = 0 ## max id of targets seen so far
blip.deltas = NULL ## linear offsets for coordinates of blips when
painting in scan and class matrices

targets = NULL ## target data frame
```

Genblips Configuration Script

```
## svn $Id: genblips.conf.R 703 2011-01-07 16:33:09Z john $
##
##   GENBLIPS  INTERFACE  CONFIG
##
##  These are parameters for the genblips interface plugin.
##  This plugin is used to define generic methods for the
##  the other interface plugins (which define class
##  methods for their object class)
##
##  DO NOT EDIT THIS FILE WHILE RUNNING radR, because
##  it is overwritten with new values when you exit
##  radR.
##
##  Lines beginning with "#" and blank lines are preserved
##  but comments at the ends of lines are not.
##
##  Make sure to include a comma at the end of each item
##  in a list.  A list definition should look like this:
##
##     what.ever = list (
##        something = 45,
##        something.else = 55
##     )
##
##  The first and last lines must not have any actual list
##  items on them, and all items must be named
##  (i.e. NAME = THING, instead of just THING)


      enabled = FALSE

## number of consecutive scans for which to generate blips
## Inf means generate until explicitly stopped

      num.scans = Inf

## generate a .CSV file with the true target tracks?

      generate.csv.file = FALSE

## should the .CSV file receive all target positions, or just visible
ones?

      only.export.visible = FALSE

## name of csv file

      csv.filename = "truetracks.csv"

### PARAMETERS DESCRIBING THE SCANS
```

```
      scan.info = list (
  ## pulses per scan

        pulses = 4096,
      #default = 1024
  ## samples per pulse

        samples.per.pulse = 1024,
      #default 512
  ## bits per sample

        bits.per.sample = 12,

  ## duration of a scan (milliseconds)

        duration = 2292,

  ## what distance increment each sample represents (metres)

        sample.dist = 4.684257,

  ## distance at which first sample begins (metres)

        first.sample.dist = 0,

  ## how many degrees clockwise from north is the first pulse?

        bearing = 0,

  ## rotation direction (either +1 or -1)

        orientation = 1
        )

### PARAMETERS DESCRIBING HOW TARGETS/BLIPS ARE PLOTTED

      radius = 4

### PARAMETERS DESCRIBING TARGETS

## maximum number of concurrent targets in dynamic model

      target.n.max = 1000

## function returning number of targets to start in this scan
## (called in each scan); called with scan number

      target.new.mu = 5

      target.num.new = expression(2 + rpois(1, target.new.mu))

## function returning radar cross sections for n target blips
```

```
      target.rcs.mu = 7

      target.rcs = expression(2 + rpois(n, target.rcs.mu))

## function returning start locations for n targets
## this should be a 3xn matrix, with locations in metres
## east of, north of, and above the radar
## These targets are deliberately started outside of the radar's range so
that only full
## tracks are generated

      target.x0 = expression({
    r <- runif(n, max.range * 1.4, max.range * 1.4)
    theta <- runif(n, 0, 2 * pi)
    z <- runif(n, 0, 1468.954671)
    cbind(r * cos(theta), r * sin(theta), z)
})
     #r <-  runif(n, 100, 100)

## function returning initial velocities for n targets
## this should be a 3xn matrix, with component velocities in metres/scan
## to the east, to the north, and up

      target.v0 = expression({
    vel <- runif(n, 7, 20)
    dir <- runif(n, 0, 2 * pi)
    z <- rep(0, n)
    cbind(x = vel * cos(dir), y = vel * sin(dir), z)
})


## function returning accelerations for n targets
## this should be a 3xn matrix, with component accelerations in
metres/scan
## to the east, to the north, and up

      target.a0 = expression(matrix(0, n, 3))

## decay for the acceleration; acceleration vector is multiplied
## by this scalar (or vector) each scan

      target.accel.decay = 0.85

## maximum speed of a target (km/h)

      target.max.speed = 500

### PARAMETERS DESCRIBING NOISE

## function returning number of noise blips to add to this
## scan

      noise.blips.mu = 0
```

```
      noise.num.blips = expression(rpois(1, noise.blips.mu))

## function returning radar cross section for n noise blips

      noise.rcs.mu = 4

      noise.rcs = expression(rpois(n, noise.rcs.mu))

## function returning locations for n noise blips
## this should be a 3xn matrix, with locations in metres
## east of, north of, and above the radar

      noise.x0 = expression(cbind(matrix(runif(2 * n, 0, 2 * max.range) -
max.range,
    nrow = n, ncol = 2), runif(n, 0, max.range/10)))
```