12 Ranging Measurements in Three-Space

The global positioning system (GPS) and some acoustic instruments provide long-baseline navigation - wherein a number of very long range measurements can be used to triangulate. We call the item that we want to track the *Target*, and the nodes in the navigation system the *Satellites*. The locations of the satellites are assumed to be well known, and what we measure during tracking are ranges from the satellites to the target. In a plane, you can appreciate that two satellites would provide two range measurements, and the target could then be located on one of two points that form the intersection of two circles.

1. For a planar setting, how many satellites are required to uniquely locate a target at an arbitrary location, and how these should be laid out? Include sketches as needed to explain your reasoning.

   Three satellites give three ranges, corresponding with three circles. The intersection of circles is - in the best case - a single point, the unique localization result. There are several notable conditions where you'll get bad results with three satellites. If they are colinear, you get no information about location in the direction perpendicular to the line. When noise is included (as below), we don't want to be even close to colinear: we want low aspect-ratio ("not too long and thin") triangles. More fundamentally, if the target is located on or near the line connecting any two satellites, then it is as if we have lost one satellite - the second range measurement does us very little good.

2. Consider a problem now in three-space. Two satellites are given, with locations [X,Y,Z] of $[500, 500, 1000]m$ and $[-500, 500, 1000]m$. The target ranges are measured at $724m$ and $768.2m$ respectively, and suppose we know also that the target is at an altitude z of less than $1000m$. Can you make an estimate for the target location $[x, y, z]$ in three-space? If so, give it.

   No, you cannot make an estimate, because you only have two constraints. Think of each range measurement as a sphere in three-space; the intersection of two spheres is a circle. Without more information, you can't say what is the $[x, z]$ *location, although the* y *location is easy to get.*

3. Augment these two satellites with third, having position $[500, -500, 1000]m$; the corresponding range measurement is $649.8m$. Can you make an estimate for the target location in three space? If so, give it.

   Yes, we have enough information now to do a full localization. See the attached code for a numerical approach. Note that at least one student figured out the answer through algebraic manipulation, despite the fact that the solution satisfies three nonlinear equations! The location of the target is $[33, -51, 950]m$. *There is another solution that will satisfy the three range equations,* $[33, -51, 1050]m$*; it is the mirror through the plane of the satellites.*

4. Almost all sensors have noise, and such range-based navigation systems are no exception. What is the sensitivity of your best computed target location above to a

one-meter error in each of the three range measurements? (Perturb the range measurements separately.) Give an explanation for what you see happening, using sketches.

Adding one meter onto each of the three ranges in turn gives solutions of $[32.3, -51.7, 950.2]m$, $[33.8, -51.0, 943.3]m$, *and* $[33.0, -50.3, 943.3]m$, *respectively. The worst error here is about* $6.7m$, *pretty bad for a one-meter range error. It occurs because the target is fairly close to the plane of the satellites; as alluded to in the planar case, these measurements provide only poor-quality information about the direction normal to the plane, and it does not stand up to noise.*

5. Find a location for a fourth satellite, that will bring down the sensitivity of the localization to noisy range measurements. Demonstrate that the whole system is now better behaved with noise, and explain why, using sketches if necessary.

   Since the problem is that the target is close to the plane of satellites, a logical solution is to put a satellite far from the plane. Indeed, if we put one at $[0, 0, 0]m$, *we create a very nice tetrahedron-like shape, and the errors from perturbations in all four channels go to* $[32.3, -51.7, 950.0]m$, $[33.4, -51.4, 949.9]m$, $[32.7, -50.7, 950.0]m$, $[33.0, -51.1, 951.0]m$. *The positioning error now is on a par with the range error, and this system is considerably more robust against noise.*

   In the algorithm, you notice that we are here solving four equations in three unknowns. As written, this is an over-determined problem and we are performing a least-squares fit: the sum of squares of the range equation errors is minimized. I use the fact that the given ranges are Euclidean distances. The MATLAB command `fminsearch()` *(synonymous with* `fmins()` *or* `fsolve()` *in some versions) does multi-variable function minimization well enough for this problem. For this, use as your error function (to be minimized) the sum of the squared errors of all the range equations.*

These questions are in the flavor of the geometric dilution of precision (GDOP) problem in GPS navigation: the performance we get from the system is very strongly related to the physical arrangement of the satellites and target. Try the Wikipedia page on GDOP.

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Three-dimensional Ranging
% MIT 2.017 FSH Sept 2009
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

clear all;
global X Y Z R ;

X = [500 -500 500 0 ] ; % Satellite locations in Cartesian space
Y = [500 500  -500  0 ] ;
Z = [1000 1000 1000 0 ] ;
```

```
x = 33 ; y = -51 ; z = 950 ; % true target location

for i = 1:length(X), % calculate the ranges
    R(i) = sqrt((X(i)-x)^2 + (Y(i)-y)^2 + (Z(i)-z)^2) ;
end;

% plot the layout of sensors
figure(1);clf;hold off;
for i = 1:length(X),
    plot3([X(i) X(i)],[Y(i) Y(i)],[0 Z(i)]);
    hold on;
    plot3(X(i),Y(i),Z(i),'o','LineWidth',2);
end;
plot3([x x],[y y],[0 z],'r');
plot3(x,y,z,'rs','LineWidth',2);
grid;
xlabel('x, m'); ylabel('y, m'); zlabel('z, m');

% apply per-channel range errors to see what happens to the estimates
R(4) = R(4) + 1 ;

[posCalc] = fminsearch('rangeError',[0 0 0]); % search for the minimum
                                               % error solution

err = rangeError(posCalc) ;                    % get the consistency error
                                               % for this solution
disp(sprintf('Solution Consistency Error:  %g.', err)) ;

% here's the error with respect to the true solution
disp(sprintf('Solution Error wrt Actual:  %g.', norm(posCalc-[x y z],2)));

% add the points to the plot
xCalc = posCalc(1) ; yCalc = posCalc(2) ; zCalc = posCalc(3) ;
plot3([xCalc xCalc],[yCalc yCalc],[0 zCalc],'m');
plot3(xCalc,yCalc,zCalc,'m*','LineWidth',2);
text(xCalc+60, yCalc, zCalc, 'Target Estimate');

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function [err] = rangeError(pos);
global X Y Z R ;
```

```
x = pos(1) ; y = pos(2) ; z = pos(3) ;

err = 0 ;
for i = 1:length(R),
    err = err + ( R(i)^2 - (X(i)-x)^2 - (Y(i)-y)^2 - (Z(i)-z)^2 )^2 ;
end;

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```