BYNING THE EARTH

Robert G. Chamberlain
Jay E. Braun
Paul J. Firnott

Jet Propulsion Laboratory
4800 Oak Grove Drive
Pasadena, California 91109-8099, U.S.A.

ABSTRACT

The Corps Battle Simulation (CBS) simulates ground and air battles between modern military forces. It runs in real time, modeling the interactions of tens of thousands of units in support of U.S. Army training exercises, typically involving several sites at geographically separated locations around the world.

Through Version 1.5.2, the CBS terrain database associated terrain characteristics with a regular grid of three-kilometer hexagons tiling an appropriate map projection of the playbook. The hexagons also served as bins for other location-dependent information to simplify the search for what's nearby.

This paper presents a new way to describe locations: Bynary latitudes and longitudes define patches of various sizes that completely cover the globe. Large patches, not necessarily all the same size, can serve as bins. Medium-sized patches can be assumed to have uniform terrain characteristics throughout. (That assumption, however, has not been used in CBS Version 1.5.3.) Tiny patches are suitable descriptors of locations.

Spherical coordinates (latitudes and longitudes) are used for fundamental specification of locations, so playboxes are not limited by the characteristics of a map projection. The natural data structure is a quadtree, which facilitates storing terrain (and other location-dependent) data in variable size bins. If desired, huge bins can be used for areas remote from the primary theater of operations and smaller bins can be used in battle zones.

1 INTRODUCTION

We have devised a new way to define locations and to describe patches of terrain for possible use in CBS: Successive bisections of the earth generate byns, coordinates that specify an area (rather than a point), establishing bynary representations of latitudes and longitudes. Byn—and its derivatives—are invented words; the y is always pronounced long. A bynary latitude is a band bounded by parallels of latitude; a bynary longitude is a zone bounded by meridians of longitude. The angular width of the band or zone is dictated by the byn's fineness. High fineness numbers imply small areas.

Byns can also define bins that cover the globe and can be used to facilitate efficient searches for what's nearby. If desired, the same procedures can be used to define medium-sized patches that can be assumed to have uniform terrain characteristics throughout. Bins may contain several terrain patches.

Byns are defined in a manner that is very natural for computers. Hence, it should be possible to develop low-level utility routines that are very efficient and compact. For example, finding the bin in which a location lies can use an efficient tree search algorithm. Creation and parsing of bynary latitudes and longitudes is all but native to digital computers, should be extremely fast, and can be completely transparent to users. Although binary representation of byns is very useful to understanding the concept and is expected to facilitate implementation, it is not requisite for validity or usefulness.

2 PARADIGM: BYNARY COORDINATES DEFINE AN AREA, NOT A POINT

Consider successive bisections of the range of possible longitudes, as illustrated in Figure 1. Construct a string of zeroes and ones—a binary number—to keep track of whether the first or second half was used to create each piece. Also keep track of how many bisections were required—the length of the string of zeroes and ones. If we store the string in the left side (up to 24 bits) of a 32-bit computer word, the fineness (the length of the string) can be stored in the final 8 bits of that word. Call this word a bynary longitude. The westward edge of this zone is obtained by interpreting the left side of the bynary longitude, the angular width by interpreting the
fineness. The resultant number can conveniently be viewed in a hexadecimal (rather than binary) format, with the fineness enclosed in parentheses. For example, the longitude byn 000000(01) refers to the entire eastern hemisphere, while 000000(18) refers to a zone that is less than one-tenth of a second of arc wide and passes through Greenwich Observatory. (Recall that 24 in decimal notation is represented as 18 in hexadecimal.)

The first bisection creates a piece, 0(1), that extends from 0°E to 180°E (the eastern hemisphere) and a piece, 1(1), from 180°W to 0°W (the western hemisphere). Bisecting the 1(1) piece creates pieces 10(2), from 180°W to 90°W, and 11(2), from 90°W to 0°W. Bisecting the 11(2) piece creates pieces 110(3), from 90°W to 45°W, and 111(3), from 45°W to 0°W. And so on.

Create a bimary latitude by similar bisections of the range of possible latitudes, with the southern hemisphere being the first piece and the northern hemisphere being the second. Latitudes, however, span only 180° of arc, while longitudes span 360°. We’d like a byn to be a binary fraction of a complete circle, so that each position in a byn translates to the same angular width. Consequently, the first bit of a bimary latitude will always be 0.

Bynary latitudes start at the South Pole, increase northward, and end at the North Pole. Bynary longitudes start at the Prime Meridian (the one through Greenwich Observatory), increase eastward, and complete the circle back at the Prime Meridian. Neither the text labels ° N, ° S, ° E, or ° W nor a mathematical sign, needed in conventional expressions of latitude and longitude, are required. The procedure is similar to boxing the compass, as illustrated in Figure 2. In both cases, the complete circle is successively bisected, but a byn defines a portion of the circle, while a compass point defines a direction.

3 PARISING: HOW TO CREATE OR INTERPRET A BYN

A byn is created by successive bisections of a circle. Thus, each position has a significance very similar to that of each digit in a binary number. In fact, a byn is a binary fraction of a circle, with an implicit binary (not decimal) point at the left. The great circle distance between two locations is simply the product of the radius of the earth, 6,371 kilometers, and the subtended central angle expressed in radians. (Of course, since the earth is not quite a sphere, any statement of its “radius” to this many significant figures is true only at a few places on the earth’s surface. Give or take 20 kilometers or so, however, this number is correct everywhere.) When the great circle arc is expressed in byns, the distance is the product of the byn and the circumference of the earth, 40,029 kilometers; conversion to radians is not needed.

To create bynary latitude and longitudes from conventionally described latitudes and longitudes: First,
Table 1: Fineness — or — Contribution to Bynary Latitude or Longitude

<table>
<thead>
<tr>
<th>p, fineness or binary position</th>
<th>radians $\frac{2 \pi}{2^p}$</th>
<th>degrees minutes seconds</th>
<th>meters (but only on a great circle)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>3.141 592 654</td>
<td>180° 0' 0&quot;</td>
<td>20,014,427</td>
</tr>
<tr>
<td>2</td>
<td>1.570 796 327</td>
<td>90° 0' 0&quot;</td>
<td>10,007,214</td>
</tr>
<tr>
<td>3</td>
<td>0.785 398 163</td>
<td>45° 0' 0&quot;</td>
<td>5,003,607</td>
</tr>
<tr>
<td>4</td>
<td>0.392 699 081</td>
<td>22° 30' 0&quot;</td>
<td>2,501,803</td>
</tr>
<tr>
<td>5</td>
<td>0.196 349 540</td>
<td>11° 15' 0&quot;</td>
<td>1,250,902</td>
</tr>
<tr>
<td>6</td>
<td>0.098 174 770</td>
<td>5° 37' 30&quot;</td>
<td>625,451</td>
</tr>
<tr>
<td>7</td>
<td>0.049 087 385</td>
<td>2° 48' 45&quot;</td>
<td>312,725</td>
</tr>
<tr>
<td>8</td>
<td>0.024 543 692</td>
<td>1° 24' 22.5&quot;</td>
<td>156,363</td>
</tr>
<tr>
<td>9</td>
<td>0.012 271 846</td>
<td>0° 42' 11.25&quot;</td>
<td>78,181</td>
</tr>
<tr>
<td>10</td>
<td>0.006 135 923</td>
<td>0° 21' 5.625&quot;</td>
<td>39,091</td>
</tr>
<tr>
<td>11</td>
<td>0.003 067 961</td>
<td>0° 10' 32.812 5&quot;</td>
<td>19,545</td>
</tr>
<tr>
<td>12</td>
<td>0.001 533 980</td>
<td>0° 5' 16.406 25&quot;</td>
<td>9,773</td>
</tr>
<tr>
<td>13</td>
<td>0.000 766 990</td>
<td>0° 2' 38.203 13&quot;</td>
<td>4,886</td>
</tr>
<tr>
<td>14</td>
<td>0.000 383 495</td>
<td>0° 1' 19.101 56&quot;</td>
<td>2,443</td>
</tr>
<tr>
<td>15</td>
<td>0.000 191 747</td>
<td>0° 0' 39.550 78&quot;</td>
<td>1,222</td>
</tr>
<tr>
<td>16</td>
<td>0.000 095 873</td>
<td>0° 0' 19.775 39&quot;</td>
<td>610.8</td>
</tr>
<tr>
<td>17</td>
<td>0.000 047 936</td>
<td>0° 0' 9.887 70&quot;</td>
<td>305.4</td>
</tr>
<tr>
<td>18</td>
<td>0.000 023 968</td>
<td>0° 0' 4.943 85&quot;</td>
<td>152.7</td>
</tr>
<tr>
<td>19</td>
<td>0.000 011 984</td>
<td>0° 0' 2.471 92&quot;</td>
<td>76.35</td>
</tr>
<tr>
<td>20</td>
<td>0.000 005 992</td>
<td>0° 0' 1.235 96&quot;</td>
<td>38.17</td>
</tr>
<tr>
<td>21</td>
<td>0.000 002 996</td>
<td>0° 0' 0.617 98&quot;</td>
<td>19.09</td>
</tr>
<tr>
<td>22</td>
<td>0.000 001 498</td>
<td>0° 0' 0.308 99&quot;</td>
<td>9.544</td>
</tr>
<tr>
<td>23</td>
<td>0.000 000 749</td>
<td>0° 0' 0.154 50&quot;</td>
<td>4.772</td>
</tr>
<tr>
<td>24</td>
<td>0.000 000 374</td>
<td>0° 0' 0.077 25&quot;</td>
<td>2.386</td>
</tr>
</tbody>
</table>

determine the desired finenesses. Next, since conventional latitudes are expressed as positive numbers of degrees north or south of the equator, while bynary latitudes start at the South Pole, increase the latitude by 90° in the northern hemisphere, or subtract the latitude from 90° in the southern hemisphere. Further, since conventional longitudes are expressed as positive numbers of degrees east or west of the Prime Meridian, while binary longitudes are all east of the Prime Meridian, longitudes in the western hemisphere must be subtracted from 360°. Computationally, the remaining two steps are trivial: Divide by 360° to get a fraction of a circle, already in binary notation if the computation is performed on most digital computers, then discard the binary point. (In decimal notation, the result of dividing 90° by 360° is represented as .25, which is two times 1/101 plus five times 1/102. The optional zero to the left of the decimal point is not a “significant” digit and has been omitted. The same quotient is represented as .01 in binary notation. That is, it is zero times 1/21 plus one times 1/22. Discarding the binary point gives the string 01 to represent the angle.)

The process performed by the computer is long division, using binary arithmetic, which can be described in this context as follows (an example follows in the next paragraph): Initialize the remainder to the latitude (or longitude) after modification as described above. The first trial subtrahend is 360°/2, or 180°. If the remainder does not exceed the trial subtrahend, the bit is 0. If it does, the bit is 1, and the trial subtrahend is subtracted from the remainder. Until the number of accumulated bits equals the fineness, repeat the process, with each trial subtrahend half the previous one.

Consider, for example, the bynary expression of the latitude and longitude of Berlin, located at 52° 31' N latitude, 13° 24' E longitude. A latitude fineness of 11 bits corresponds to 10.5° of arc, a bit under 20 kilometers in north-south extent. Since a degree of longitude is only 60% as long at Berlin’s latitude, fewer bits are needed in longitude: 10 bits produce an east-west extent a little under 24 kilometers. This is a reasonable precision for a city the size of Berlin. Table 2 illustrates the calculation. When the number in the “trial subtrahend for bynary digit” column is larger than the current “remainder”, make the bynary digit 0 and go to the next row. Otherwise, make the digit 1 and reduce the remainder accordingly. The resultant bynary latitude for Berlin is 01100101010(1011) or, in the (perhaps) more
Table 2: Calculation of the Bynary Latitude and Longitude - An Example

<table>
<thead>
<tr>
<th>latitude</th>
<th>remainder</th>
<th>trial subtrahend for binary digit</th>
<th>longitude</th>
<th>remainder</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td></td>
<td>180°</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>52° 31’</td>
<td>90°</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>7° 31’</td>
<td>45°</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td></td>
<td>22° 30’</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td></td>
<td>11° 15’</td>
<td>1</td>
<td>2° 9’</td>
</tr>
<tr>
<td>1</td>
<td>1° 53.5’</td>
<td>5° 37.5’</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td></td>
<td>2° 48.75’</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>0° 29.125’</td>
<td>1° 24.375’</td>
<td>1</td>
<td>0° 44.625’</td>
</tr>
<tr>
<td>0</td>
<td></td>
<td>0° 42.1875’</td>
<td>1</td>
<td>0° 2.4375’</td>
</tr>
<tr>
<td>1</td>
<td>0° 8.03125’</td>
<td>0° 21.09375’</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td></td>
<td>0° 10.546875’</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>0° 5.2734375’</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

readable 32-bit hexadecimal form, 654000(0B), with 0s inserted in the undefined 13 bits. The bynary longitude is 098000(0A).

Note that the remainders were truncated, rather than used for rounding. This is a consequence of the fact that the bynary latitudes and longitudes define bands and zones, not points: the number 654000(0B) says that Berlin lies between 52° 22’ 58.125” N latitude (the southern edge of the band) and 52° 33’ 30.9375” N latitude (the southern edge of the band plus its width) and 098000(0A) says Berlin is in the zone bounded by 13° 21’ 33.75” E longitude (the western edge of the zone) and 13° 42’ 39.375” E longitude (the western edge of the zone plus its width). The selected finenesses can be interpreted as expressing the uncertainty in our knowledge of the location of Berlin, as the resolution of our statement of that location, or as our statement of the size of Berlin.

To interpret a bynary latitude or longitude, multiply the byn by 360° to get the latitude angle northward of the South Pole or the longitude angle eastward of the Prime Meridian. This multiplication is equivalent to summing the numbers from either of the above tables that correspond to each position in the byn that contains a one, up to the fineness of the byn. The result is the parallel of latitude or the meridian of longitude that corresponds to the southern or western edge of the patch. The angular width of the patch is given by the row of the table corresponding to the fineness. The same procedure can be used to interpret the central angle of any great circle arc; multiplication of the byn by the circumference of the earth, expressed in any units, gives the length of the arc in the same units.

4 BINS AND PATCHES

In all versions of CBS to date, three-kilometer hexes that tile the playbox have been used as "bins", owning lists of various simulation objects that occupy or are associated with the hex. These lists have permitted efficient searches for "nearby" objects. Until CBS 1.4, the hexes (or their triangular sectors) were the lowest level of resolution played, so it was convenient to assign terrain characteristics to them. As fidelity in CBS has improved, resolution has increased. Bins that are sized for efficient searching are too large to be considered to have uniform terrain. Thus, the bins could contain lists of terrain patches as well as of other simulation objects. Bins and terrain patches can be defined by the same process, carried to different levels of fineness.

The two byns used to define a terrain polygon or a bin do not have to have the same fineness. These patches can be rather trapezoidal, triangular, or even circular if close enough to one of the poles. {Aside: A hunter leaves camp, travels 1 kilometer south, wanders 2x kilometers east, shoots a bear, drags it 1 kilometer back to camp. What color is the bear?} In fact, an area defined by lines of constant latitude and constant longitude is not, strictly speaking, a spherical polygon of any sort, as the parallels of latitude are not "straight lines" (that is, great circles). Almost everywhere, however, these distinctions are inconsequential and patches are virtually indistinguishable from ordinary rectangles.

Byns that define different bins do not have to have the same fineness. Longitude byns will often have smaller finenesses than latitude byns, because the length of a degree of longitude depends upon the latitude. At the equator, two 16-bit byns define a square patch 610.8
meters on a side. In Tokyo, at 36° N latitude, the patch
defined by two 16-bit byns is a rectangle 610.8 by
495.25 meters.

Remote regions might be represented by seven bit
byn (which would correspond to 100,000 square kilo-
14 meters at the equator) or even fewer. Near the equator, four-
teen-bit byns are required to define a patch with about the
same area as a three-kilometer hex.

How small can a patch defined by byns be? An
equatorial patch specified by two binary coordinates,
each of 24-bit fineness, has an area of only 5.7 square
meters—about the same as a jeep. (A HMMWV is
larger; so is a tank; so are most units). Away from the
equator, maximum fineness defines even smaller areas.
At a pole, maximum fineness in both byns defines an
area of only 10⁻⁶ square meters (a triangle 2.386 meters
in the N-S direction, 0.892 micrometers in E-W extent).
There might be other applications that would require
smaller bins, but not CBS.

5 LOCATIONS

In this paradigm, locations are areas, not mathematical
points, but the areas implied by 24-bit byns are suffi-
ciently small to be used as if they were point locations
in the CBS context. Implementation of this idea would
not, however, paint CBS into a corner. A few changes in
low-level utility programs could improve the equa-
torial worst-possible resolution from 5.7 square meters to
0.54 square inches. (The finest resolution achievable
with 24 bits is 0.07725 seconds of arc, producing a max-
imum location uncertainty of 5.7 square meters.
Locations could be defined by all 32 bits, with 32-bit
fineness assumed (rather than specified). The cost would
be that locations and bins would not be defined and inter-
preted in the same ways. The benefit would be that loca-
tions could be 256 times as precise (to within 9.3 mm).
Those low-level routines could know the two uses apart
when parsing. So that hexadecimal notation could be
used, we decreed above that 8 bits would be used for the
fineness. Only 5 bits are actually needed, so one of the
extra bits could be used as a flag. (The value 0.54 square
inches assumes 31-bit byns.) Another alternative would
be to give up the efficiency resulting from storing the
fineness in the byn and use additional words for fineness.
As is of ten the case, trades between running time, stor-
age requirements, and programming time are available.

CBS should translate from its native format into a
user-friendly format whenever it is about to communicate
with the user—but only then. It should not be necessary
to update every variant way of expressing location data
whenever a unit moves. One implementation would be
to treat any location as an object, whose only data at-
bributes are the bynary latitude and longitude. Other
formats, such as the U.S. Military Grid Reference
System, can be produced by function attributes only
when they have to be displayed.

6 SHARING DATA

It is inevitable that many terrain patches will be de-
scribed by identical values of all terrain parameters, ex-
cept for elevation, because there are fewer combinations
of possibilities than there are patches of terrain.
Therefore, the large number of patches can be defined ef-
ciently, with only a few attributes: pointers to profiles
that contain the terrain data.

The representations of roads and rivers in CBS 1.5.3
are also expected to use extensive data sharing. In that
context, for example, many successive links in a road
may share characteristics such as size, flow, condition,
and perhaps even congestion.

7 GIVEN A LOCATION, FIND THE RIGHT
BIN AND THE RIGHT TERRAIN DATA

Although it is still premature to fully design the data
structure to be used, certain options immediately present
themselves, due to the “double binary” nature of the
scheme. For example, a quadtree (Mark 1986) can be
constructed by sequentially dividing the world into
quarters, produced by halving both the latitude and longi-
tude simultaneously. Some of these quarters are sub-
sequently divided further. When the patches defined during
this process reach the finenesses chosen for bins, stop
the process of division. Nodes on this tree have bimary
coordinates and four pointers to child nodes (or to bins).

Suppose, for example, the earth is to be represented
by bins A through K as illustrated in Figure 3. The cor-
responding quadtree is given in Figure 4. Finenesses are
expressed by the number of binary digits shown; explicit
representation of the finenesses has been suppressed in an
attempt to clarify the illustration.

Bins have bimary coordinates and may contain lists
of terrain patches, units, road and river network nodes and
links, and other simulation objects that are located in (or
otherwise relevant to objects that are located in) the bin.
The bynary coordinates of terrain patches and locations of
simulation objects are produced by the same byning
process as was used to develop the bin tree, but these
other objects are not bins. Terrain patches would have
pointers to terrain profiles. Such other objects as
minefields and antitank ditches also might use profiles to
share data.

Suppose a unit is located at coordinates 010001110,
010111000. What bin is it in and what mobility condi-
tions will the unit find? The first pair of bits, (0,0),
directs us to the node labeled 0.0. The next pair of bits,
Figure 3: Byning a Latitude-Longitude Map. (This projection is also known as *Equidistant Cylindrical.*)

(1,1), lead to node 01,01. Then to node 010,010. The 0,1 child of this node is bin G at 0100,0101. Thus, the location is in bin G. Mobility patches associated with the bin could be found by continuing the tree search (if the data is organized that way) or by stepping through bin-specific lists.

ACKNOWLEDGMENTS

The work described herein was carried out by the Jet Propulsion Laboratory, California Institute of Technology, and was sponsored by the U.S. Army Simulation, Training, and Instrumentation Command through an agreement with the National Aeronautics and Space Administration.

REFERENCE


AUTHOR BIOGRAPHIES

BOB CHAMBERLAIN is a Senior Member of the Technical Staff at the Jet Propulsion Laboratory in Pasadena, California. He received his B.S and M.S. in Mechanical Engineering from the California Institute of Technology and is a registered Mechanical Engineer. His interests have concentrated on the application of quantitative analysis to system design and development. He is co-author of the forthcoming NASA Systems Engineering Handbook and is a contributing editor to the InFORMS Journal Interfaces.

JAY BRAUN is a Simulation Software Specialist, under contract at the Jet Propulsion Laboratory. He received his Master's degree in Information Systems at UCLA. His interests are in the application of software engineering principles to simulation models, including design techniques, open systems architecture, and computational performance. He has spoken about these issues at various simulation conferences. Mr. Braun is also an on-call simulation consultant to Toyota Motor Sales, U.S.A., Inc.

PAUL FIRNETT is a Member of the Technical Staff at the Jet Propulsion Laboratory. He received his M.S. in Applied Physics from the University of California, Los Angeles. He is currently assigned as a senior programmer to the Corps Battle Simulation program for the U.S. Army.
Figure 5: Structure of the Bin Tree—Another View. This also shows the structure of the bin tree, with an emphasis on the fact that the child pointers of nodes sometimes point at another node (shown as transparent) and sometimes at a bin (shown as opaque and dotted). Note the geometric proximity of the descendants of any particular node. Nodes are shown as fuzzy-edged oblique squares.