# Novel Georeferencing Approaches Inspired By Distributed Ledger Geostamping

Abstract: GeoGnomo (GeoGnomo: https://www.geognomo.com) is an open-source project exploring various forms of georeferencing for use in mutual distributed ledgers (MDLs, aka blockchains). The project encompasses four different approaches designed to provide effective methods for georeferencing: Quaternary Triangular System, Quaternary UTM System, Quaternary Rectangular System, and Variable Rectangular System. In this paper, we examine the four approaches to georeferencing and define criteria to assess their performance. The details of each approach are presented, along with worked examples to illustrate how georeferences are generated from latitude and longitude coordinate inputs for areas. Following this, we analyse and compare the strengths and weaknesses of each system and suggest areas for further study. We conclude that Variable Rectangular System is of especial use in geostamping for mutual distributed ledgers.

Keywords: Georeferencing, Geostamping, Timestamping, Mutual distributed ledgers, Blockchain, Postcodes, Zipcodes.

## Introduction – Timestamping and Geostamping

Mutual distributed ledger (MDLs, aka blockchain) technology provides an immutable record of transactions that is shared in common and stored in multiple locations (Mainelli and Smith, 2015). MDLs are increasingly proposed as a solution to many issues in multi-organisation processing, particularly within identity, document, and agreement exchange, such as within supply chains, financial services, or identity systems. MDLs use techniques from cryptography to create tamperproof records distributed across multiple computers with little central control (Mainelli and Manson, 2016). Smart contracts are “the implementation of contract terms as executable computer code” (Mainelli and Manson, 2017). Their popularity has grown, as people realise that the computer code can be embedded in MDL technology. ‘Smart ledgers’ are MDLs with embedded, executable code. Smart ledgers are able to specify rules about the use of data within the MDL, for example, “release this ship’s location four hours after it has been recorded on the MDL”.

One of the primary functions of MDLs is to ‘timestamp’ new transactions, i.e., to provide certainty of a digital event. A file hash is a one-way algorithm that takes a digital file and creates a digital signature of a specified length. Digital events are often recorded as a 'hash'. For example, a 1 MB photograph might be recorded as a hash of "b0544cb7d5ad8d722f7a9fe15f44f7214a95acead2978e837f926fed451b182e". Anyone processing the 1 MB photograph through the same algorithm will get the same hash. A small change to the photo, just a single pixel, should produce a completely different hash, e.g., "894b72b219fb646abe9c441833aa73d77e1f419a5b2eedd57eac40e3ce44186f". However, the hash is one-way, i.e., it is impossible to recreate the photograph from the hash.

Equally, there is a similar concept of the 'geostamp', i.e., recording the location where a digital event occurs. GPS and associated technologies allow devices to record and timestamp their own positions. By connecting to a MDL, this information can be reliably used by third parties sharing the geostamp. For example, an insurer could track individual shipping containers over a multi-modal journey, recording their location as they travel, enforcing exclusion zones, and, dynamically, adjusting premia. The GeoGnomo project has timestamping and geostamping applications for clinical assessments. In 2016, 8.2 million clinical assessments were 'stamped'. In 2017, this figure had risen to 15 million. Figure 1 shows a sample heatmap of the assessments (see also https://metrognomo.com/heatmap/assessments/).

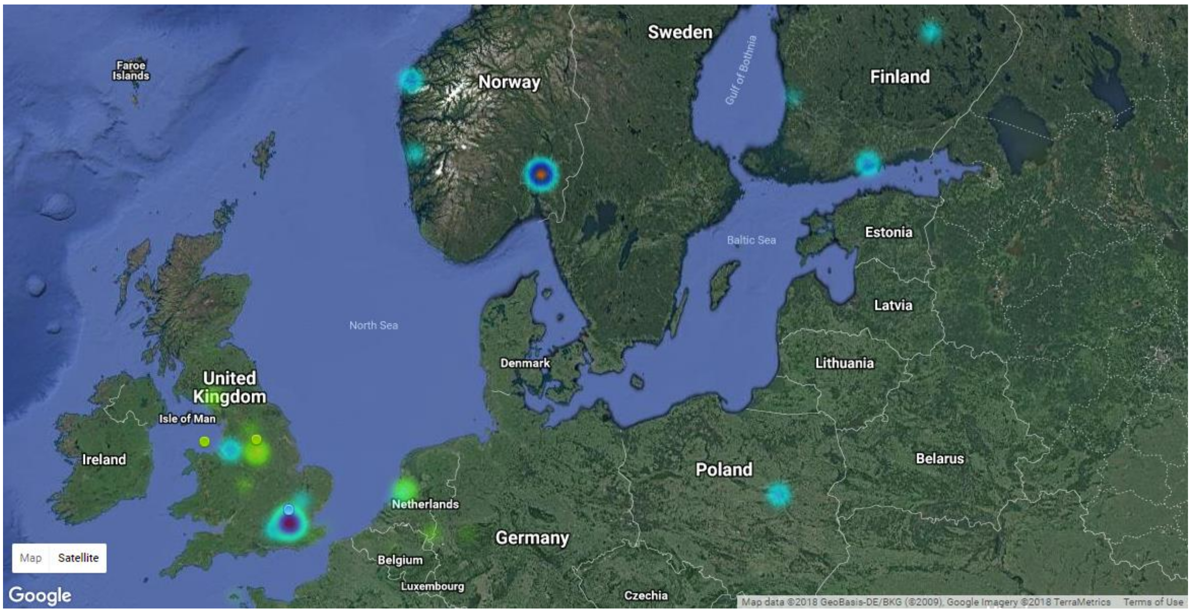



Figure 1. Sample Heatmap of Clinical Assessments

The paper will start with a brief background on georeferencing methods followed by an introduction to the Geognomo project. The four georeferencing methods will then be examined in the following order: Quaternary Triangular System (QTS), Quaternary UTM System (QUTMS), Quaternary Rectangular System (QRS) and Variable Rectangular

System (VRS). Some ideas for future consideration will be followed by a comparison of the four methods and latitude/longitude point coordinates, leading to the paper's conclusions.

## 1. Background

Being able to put names or labels of some kind on spatial locations is a very old property of human cognition, perhaps almost as old as language itself. Some non-human primates also have cognition of spatial location (Hauser et al. 2002, Egnor et al. 2005), as do some non-primates. For example, communicating spatial location by direction and distance is a property of the communication system of honey bees (Riley et al. 2005, building on work of von Frisch).

Systematically locating places became possible with the work of the Greek geographer Hipparchus, who used the system of parallels of latitude north and south of the equator and great circle meridians of longitude meeting at the poles to define coordinates. Ptolemy built on Hipparchus' work to develop what we would, perhaps, now call a gazetteer of the locations of places by their coordinates. It is possible that other cultures, also, invented similar or other methods of systematically defining locations.

Over many centuries in the European and, then, Western Hemisphere cultures, the technology for measuring latitude, and particularly longitude, improved immensely, until by the 19th Century, relatively accurate maps could be made. Although the over two-millennial old system of geographic coordinates became the primary scientific method for describing location, many other methods, such as natural language place names, continue to be in use within smaller portions of the earth's surface.

The burgeoning of georeferencing technology became much more sophisticated in the second half of the 20th Century, with the development of computers and additional locational devices based on satellite measurements. Meteorologists were among the early users of computers to model physical processes on a global basis and develop global tessellations with locational addressing to house that modelling (e.g., Sadourny et al. 1968, Williamson 1968).

In recent decades, many new global grid systems with location addressing have been developed. Designing a system of point or polygonal areas on a sphere requires a georeferencing system to be designed, so that locations can be addressed. Global grids can be created:

(1) from the quadrilaterals (and triangles at the poles) of the geographic coordinate system of latitude and longitude,

(2) from projections of the geographic coordinate system,

(3) by subdividing the faces of one of the polyhedral Platonic or Archimedian solids fit to a sphere or ellipsoid, or

(4) from polygons formed by grids of points, where the points may come from

(4a) a set of application specific observation locations, or

(4b) an algorithmic point generating process.

The latitude and longitude coordinate system, for example, has been widely used in global climate models, where special treatment is needed at the poles. Georeferencing the sub-polar latitude-longitude rectangles can be done by using the coordinates of a corner or midpoint of a rectangle at an appropriate precision for the application, and, for some applications, by combining the two coordinates into a single georeference.

An example of using a map projection to create a global grid is that of (Tobler and Chen, 1986), in which the Lambert equal-area cylindrical projection was modified to fit the earth in a square so that a quadtree data structure could be used for storing and accessing data. A polyhedral approach using hierarchical subdivisions of the eight triangles of the octahedron is that by (Dutton, 1989), with a system of location addressing in (Dutton, 1990).

A grid system created from the Voronoi polygons for a set of points given by a set of observations or instrument locations, for example, is that of (Lukatela, 1987). Georeferences could be the coordinates of the observations or the coordinates of the instruments. An example of generating a set of points by a mathematical algorithm to create Fibonacci spirals of points across the sphere is (Swinbank and Purser, 2006), with Voronoi polygons of these points forming a tessellation. Georeferences could be a numbering system for the points or could use the “zone number” defined in the article.

Chinese scholars have published a large number of articles in this century; their work appears to be oriented to polyhedral approaches, with many articles using the icosahedron. (Tong et al., 2010), for example, describe a method for addressing hierarchical hexagonal grids created by subdividing the triangles of an icosahedron (which would, also, work on the octahedron and tetrahedron).

Independently of the global grid school of work, a number of georeferencing schemes have been developed. Government agencies develop georeferencing systems for their work. The US Census Bureau, for example, manages a number of georeferencing systems for their data (https://www.census.gov/geo/reference/geocodes.html, accessed 2017-1123). The OpenStreetMap community uses a system of "quad tiles" for storage and addressing of their data (http://wiki.openstreetmap.org/wiki/QuadTiles, accessed 2017-1123). A number of different systems are listed at (https://en.wikipedia.org/wiki/List_of_geocoding_systems, accessed 2017-1123).

One system that is relevant to the approach presented in this paper is Geohash (https://en.wikipedia.org/wiki/Geohash, accessed 2017-1123). Like the GeoGnomo approach, Geohash generates georeferences as short sequences of letters and numbers to refer to latitude-longitude quadrilaterals (or triangles at the poles).

Because there is a large body of literature ever expanding in the area of global grids and georeferencing for the globe, Appendix D contains a bibliography of some of this literature.

## 2. GeoGnomo

Georeferencing is the assignment of a unique identification to a place on the earth represented by a point, line, area, or complex object. Georeferencing can use coordinate identifications such as latitude and longitude, or projected two-dimensional or three-dimensional coordinates, or other systems such as street addresses. GeoGnomo (GeoGnomo: https://www.geognomo.com/) is an open-source project designed to provide quick, easy, and effective methods for geostamping (assigning a georeference to an entry in a mutually distributed ledger system), aiming to improve on the weaknesses of the existing latitude/longitude coordinate system. We believe the principal qualities of a good georeference are:

- Memorability - it should be compact and memorable;
- Aggregation - a coding system should be able to describe a variety of shapes and structures, both natural and human, such as forests, beaches, buildings, sports grounds, country borders etc.;

- Proximity - similar codes should represent similar locations, so that people exchanging codes can roughly understand the distance and relationship between them;
- Scale - users should have control over the precision.

Geostamping can be as simple as recording a point of latitude ($\phi$ and longitude($\lambda$). However, more complicated recording, say an event occurring within a postcode or a zipcode, requires either ancillary tables for the geographic area, or recording a series of points. Areal entries range from "something occurred at this factory" to "here is the footprint of a ship travelling the world, and these events occurred at these locations". Retrieving data from an MDL based on areal intersections becomes computationally intensive if it has not been recorded in a structured manner. "What happened at this 100m by 100m location over the past year? 237 aircraft flew overhead; 10 014 cellphones were recorded; 3 robberies were reported". Further, if humans are involved, for example a care worker needing to state they attended a specific location, a memorable global 'postcode' or 'zipcode' would be useful.

GeoGnomo is proposing several systems that (1) improve georeferencing memorability, (2) through the use of area codes rather than point codes, have less variation in area than is present in one by one-degree blocks in latitude and longitude, (3) provide better aggregation than latitude and longitude coordinates, (4) maintain proximity relationships, and (5) allow control of spatial scale. The adoption of a limited number of georeferencing structures that have global applicability should ease one area of inter-operability for MDLs, namely sharing geospatial information.

## 3. Four Methods Of Geostamping

GeoGnomo has developed four methods (Quaternary Triangular System (QTS), Quaternary UTM System (QUTMS), Quaternary Rectangular System (QRS) and the Variable Rectangular System (VRS)) for georeferencing that will be presented in turn. Each method will be explained and examples will be given. Following this, the systems will be analysed and compared with one another according to the criteria set out above, before setting out a conclusion.

### 3. 1 Quaternary Triangular System (QTS)

***Method***

The Quaternary Triangle System divides the globe into a fixed grid of triangles and assigns a unique georeference to each triangle. Codes are generated from a latitude/longitude coordinate pair and a specified level $n$, which determines the scale of the grid. The code generated describes an area that contains the specified point.

The proposed grid system starts with an inscribed icosahedron in a sphere. The icosahedron is one of five platonic solids, meaning that each face is regular and identical. As a result, each face covers an equal area of the globe when modelled as a sphere; this is an important advantage over any rectangular grid where the poles become lines. Since the icosahedron has the highest number of faces of all platonic solids, a model defined on it produces partitions with less distortion than those based on the other platonic solids (White et al. 1998), although the octahedron (Goodchild et al. 1991) is also commonly used, as it can be aligned easily, so that it is symmetrical about the equator. We note that this approach, as with all the others we propose, does not create equal grid cell areas across the entire spherical globe.

We orient the icosahedron so that the vertices are at the North and South poles. Instead of directly using a radial projection onto the icosahedron, we project the globe onto the two-dimensional faces of the icosahedron. Our projection is a hybrid of two methods, an equirectangular projection for latitudes near the equator and a variation of the Collignon projection (Goodchild and Shiren 1992) for latitudes nearer the poles, which helps to reduce distortion. We then start a recursive division process, where each triangle facet is decomposed into four equilateral triangles in the plane of a face of the icosahedron, called its ‘children’. Figure 2 shows how this divides the globe at the first level of decomposition.

Figure 2. The Quaternary Triangular System (QTS) Grid with one level of subdivision

The four child triangles are labelled 0 to 3 according to Figure 3. At the n-th level of decomposition, we have an n-digit quaternary sequence, preceded by a base 20 digit representing the initial decomposition at level 0. We call this a quaternary code or quaternary trail $G$, so we have

$$G = a_0, a_1 a_2 a_3 \dots a_n$$

Where $a_0$ indicates which of the 20 faces the point lies within and $a_1 a_2 a_3 \dots a_n$ implies the zooming in 'path' taken when dividing the triangles. These triangles can be either “upward” or “downward” and following (Goodchild and Shiren 1992). We reflect the labelling rules when the orientation changes, avoiding unnecessary complications to the calculations, as shown in Figure 3.  Therefore, following the zooming in ‘path’, a sequence of 2s will point you towards the left corner of the triangle.

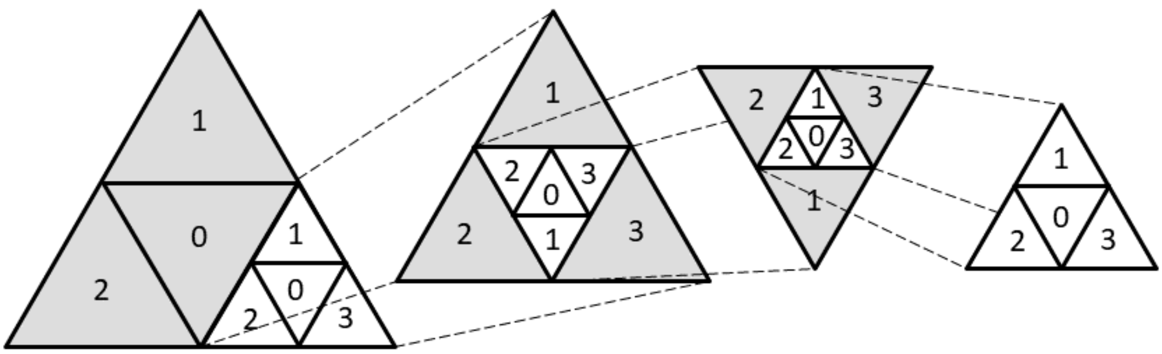



Figure 3. Triangle labelling rules showing child triangles

Given latitude and longitude coordinates, the quaternary trail is established through a series of calculations found in Appendix A. In order to express the quaternary code in a more compact way, we encode it in base 32 using an alphabet of the twenty-four letters A–X (or phonetic alphabet Alpha to Xray) and eight digits 2–9 (0 and 1 are excluded to avoid confusion with O and I respectively). The level is attached to the end of the code.

***Examples***

The main example given for each system will be a map view of central London, with an approximation of the respective grids, showing codes at multiple levels. Figure 4 shows the map for QTS. This illustrates the relationship between area codes that are close together, which is maintained over all levels due to the method we have used for base 32 encoding.

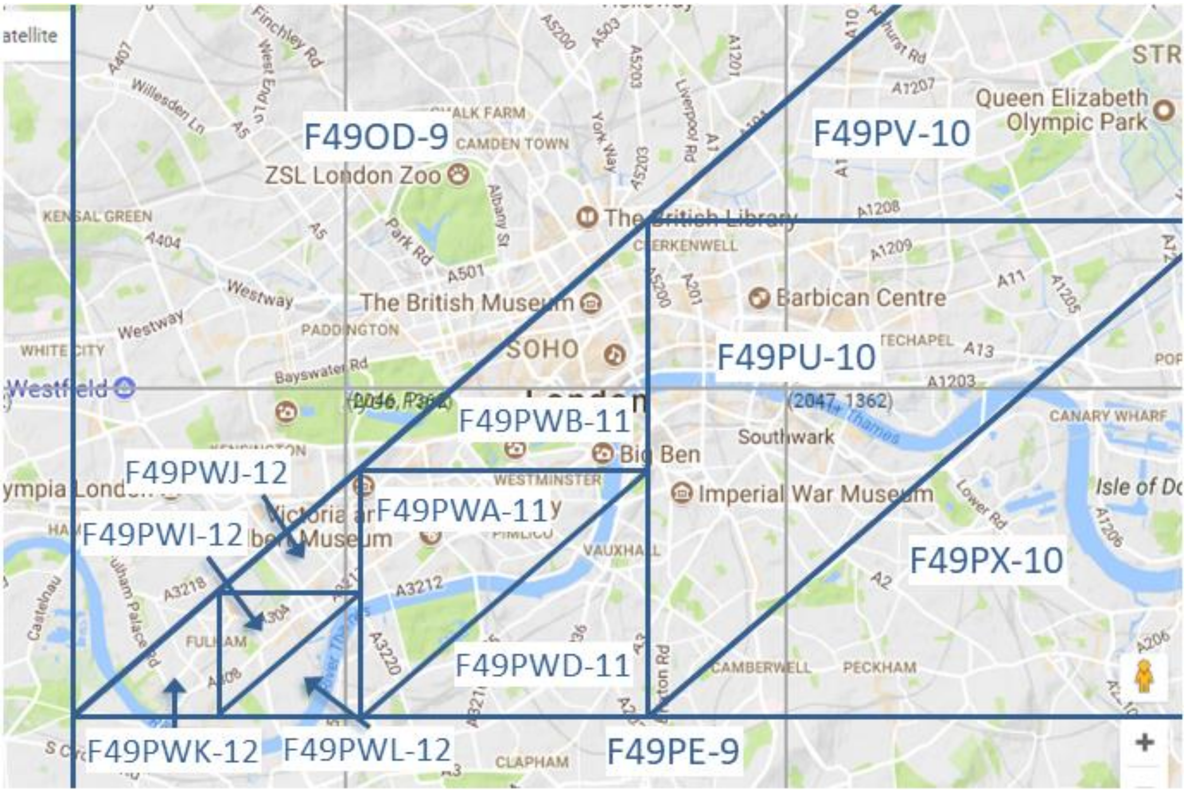



Figure 4. The QTS grid over central London

Note: Geognomo now requires “QTS:” entered before the alphanumeric code in order to find a location.

A second example will illustrate the process of converting the quaternary code into the base 32 code in slightly more detail, as it is, also, relevant to the other systems. We consider Big Ben, described by the latitude and longitude coordinates (51.500732, -0.124626). A level 14 quaternary code describes a triangle with side lengths of roughly 500m. In this case, the quaternary code is calculated to be:

$$5,31133133121332$$

The above quaternary code has one base 20 digit and 14 quaternary digits. To convert this into base 32 notation, we first encode the quaternary code into binary:

$$101,1101011111011111011001111110$$

Then zeros are padded onto the right of the first number to make a group of 5 bits. Following this, bits are arranged into groups of 5 from left to right. In the event that the last group is not a group of 5, zeros are padded to the right of this group. Using this method ensures that the code changes as little as possible when changing levels:

$$00101,110101111101111101100111100110$$

The groups of 5 are then converted to base 32 using the base 32 alphabet. In order to decode this, we must also attach the level to the end of the code, so that the decoder knows how many 0s were added to the last group of 5 bits. Therefore, we have the following GeoGnomo code:

$QTS: F49PWPG - 14$ or $Foxtrot, 4,9, Papa, Whiskey, Papa, Golf - 14$

To specify an area that corresponds to Big Ben itself, we increase the level to 19, resulting in a precision of about 15m. Notice that the code is the same as the level 14 code, until the last 3 characters, 2 of which have been added to increase the precision and only the last character of the level 14 code has been changed:

$QTS: F49PWP23A - 19$or $Foxtrot, 4,9, Papa, Whiskey, Papa, 2,3, Alpha - 19$

Figure 5 shows the level 14 triangle on the left and the level 19 triangle on the right. In each case the area corresponding to the code itself is red, while the blue triangles represent its neighbours.


Figure 5. Big Ben shown at level 14 (left) and level 19 (right)

***Grid details***

Appendix C contains calculations concerning the corners, neighbours, and sizes of the triangles at each level. Table 1 below summarises the sizes at various levels. At level 23, the side lengths are less than 1m and by level 37, the triangles are smaller than the unaided human eye can see.

| Level | Number of Triangles | Side length along equator (m) | Average Area ($m^2$) |
|---|---|---|---|
| 1 | 80 | 4 007 501.7 | 6 375 900 000 000 |
| 2 | 320 | 2 003 750.9 | 1 593 975 000 000 |
| 3 | 1 280 | 1 001 875.9 | 398 493 750 000 |
| 4 | 5 120 | 500 937.7 | 99 623 437 500 |
| 5 | 20 480 | 250 468.9 | 24 905 859 375 |

| | | | |
|---|---|---|---|
| 10 | 20 971 520 | 7 827.2 | 24 322 128.3 |
| 15 | 21 474 836 480 | 244.6 | 23 752.1 |
| 20 | 21 990 232 555 520 | 7.6 | 23.2 |
| 30 | 23 058 430 092 136 900 000 | 0.0075 | 0.000022121 |
| 40 | 2.41785E+25 | 7.2896E-06 | 2.10961E-11 |
| … | …. | … | … |
| 140 | 3.88534E+85 | Side length less than the Planck length! | |
| n | $20 * 4^n$ | $\frac{2\pi R}{5 * 2^n}$ | $\frac{4\pi R^2}{20 * 4^n}$ |

Table 1. Triangle dimensions for QTS with Radius 6378.1km along Equator (https://en.wikipedia.org/wiki/Earth).

Note that the displayed grid is an approximation. Lines that do not run parallel to latitude or longitude lines should not be perfectly straight, due to the distortion introduced by the Mercator projection used by Google Maps. The distortion is minimal near the equator, and so the straight lines are, in fact, a very accurate approximation. However, at higher latitudes there is extra distortion introduced by the projection method we have used, and large triangles may be inaccurately plotted on the map. The effect of this is reduced as the zoom level is increased, so for smaller triangles the approximation is still accurate.

***Trace***

We define the 'trace' as the path of locations given by the list of possible codes sorted in alphabetical (with respect to the base 32 alphabet) order. Figure 6 shows the trace given by the QTS within each level 0 triangle with our chosen labelling system, starting from the middle. A trace represents the path from one grid cell to its nearest neighbour in the coding system. Knowing the trace of the curve allows us to better visualise how neighbouring triangles have similar codes, and when they don't (i.e. they are on the edge of a higher-level triangle) how different the code is likely to be based on how far the trace still has to go. The space filling curves are chosen to keep numbering at all levels consistent.

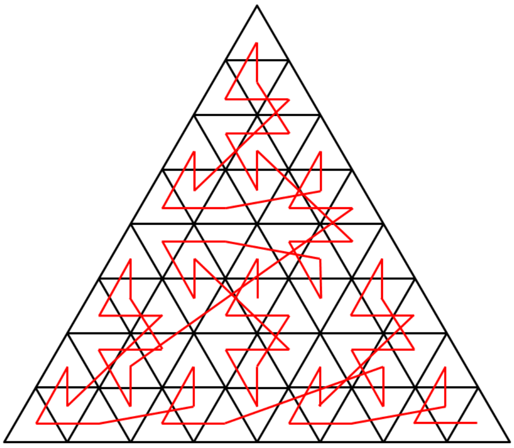


Figure 6. The trace within each level 0 triangle

The trace depends on the sub triangle labelling shown in Figure 3. There is only one other possible trace, produced by moving 1 to the middle and replacing it with 0, as any other trace is a rotated or reversed version of one of these two. We believe assigning 0 to the middle sub triangle is more intuitive than assigning 0 to a corner triangle.

### 3. 2 Quaternary UTM System (QUTMS)

***Method***

Quaternary UTM System uses the same quaternary trail method as QTS but defines its level 0 grid according to the Universal Transverse Mercator (UTM) projection, combined with the Military Grid Reference System (MGRS). It, also, generates codes from a single point and a specified scale level, with the code representing the area that the point lies in.  Figure 7 shows how the level 0 grid divides the globe.

Figure 7. The Quaternary UTM System (QUTMS) Grid at level 0

The UTM projection is constructed over 60 longitude bands of equal width. Each band has a centre, on which the globe is projected using a secant transverse Mercator projection, ensuring minimal overall distortion within each band. The bands are numbered from 1 to 60, with the left edge of zone 1 starting at 180W. Latitude bands are defined by the MGRS; bands between 80S and 84N are assigned letters from C-X (I and O omitted) and are 8 degrees tall, apart from band X, which is 12 degrees tall. Together, a number and letter represent a rectangular grid zone.

However, there are some exceptions. Firstly, 4 polar regions are labelled A, B, Y and Z. The South Pole is covered by A (west) and B (east) and the North pole is covered by Y (west) and Z (east). The region 32V, corresponding to part of the coast of Norway, is larger than normal, and 31V is smaller, so that 31V covers only water. Furthermore, the regions 32X, 34X, and 36X are not used, and are instead covered by enlarging 31X, 33X, 35X and 37X. Using these rectangular grid zones to form the level 0 grid, we then establish a quaternary trail. Because the edges of the rectangles lie parallel to longitude and latitude, the calculations are simplified from QTS and the algorithm is shown in Appendix B. The labelling for sub rectangles is shown in Figure 8 below:

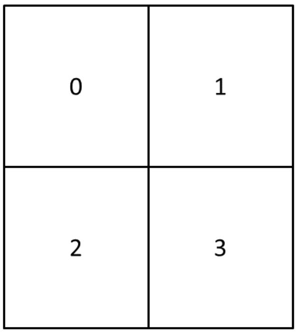



Figure 8. Labelling rule for QUTMS

After the quaternary trail has been established, it is encoded in base 32, but the level 0 rectangle is left in its number-letter form, so that the UTM zones can be understood directly form the codes. Because the number part represents longitude and the letter part represents latitude, it is easier to understand the approximate part of the world that the area is in.

***Example***

Figure 9 shows the QUTMS grid over central London. The codes are very similar to QTS codes in proximity and memorability. Due to London’s latitude of around 51N, the grid rectangles appear very long and thin on the map. This illustrates the case for using a platonic solid in QTS to maximise consistency over the entire grid.

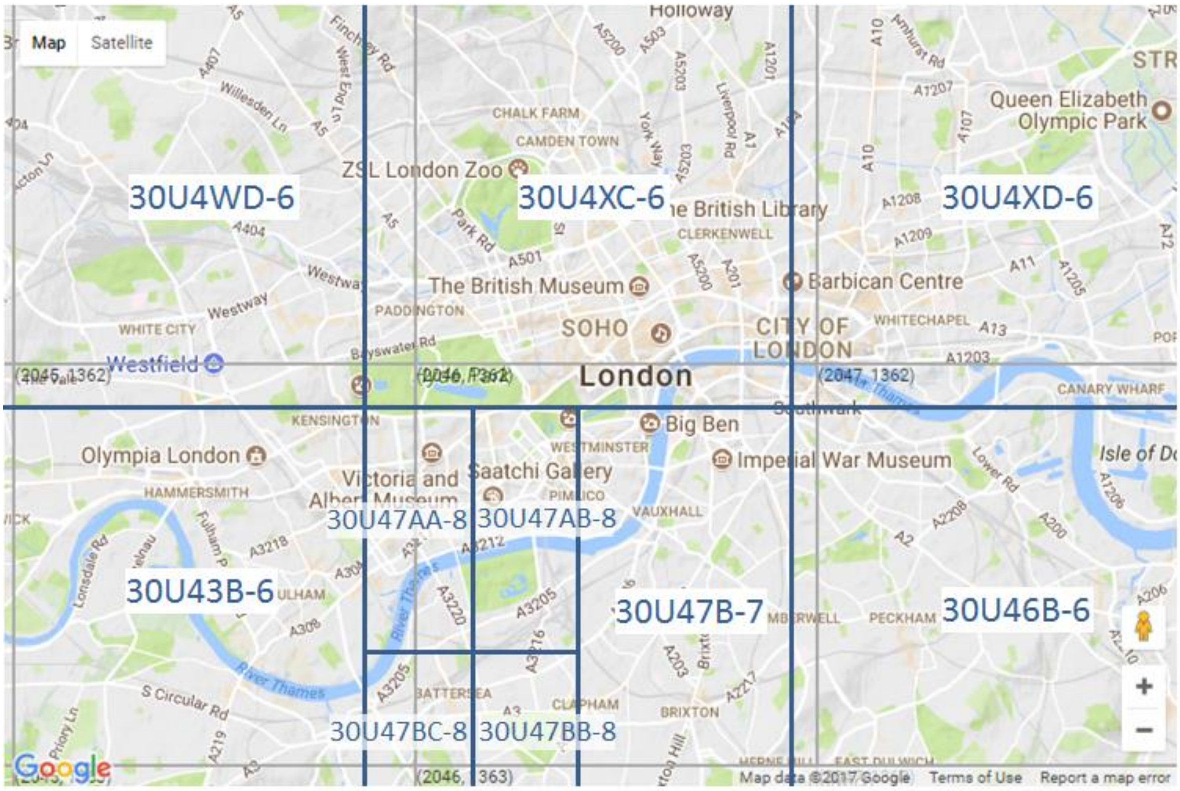



Figure 9. The QUTMS grid over central London
Note: Geognomo now requires "QUTMS:" entered before the alphanumeric code in order to find a location.

### *Grid details*

In addition to triangle details, Appendix C includes the general formula for areas of a rectangle (or formally, a quadrangle) on the globe. Table 2 summarises the grid areas at different levels at the equator and at row X. This quantifies the traits seen in the above example, showing that the reduction in width of the rectangles near the poles reduces their area to around 1/3rd of the equivalent rectangle at the equator.

| Level | Side length along equator (m) | Side length along row X (m) | Area at equator ($m^2$) | Area at row X ($m^2$) |
|---|---|---|---|---|
| 1 | 333 956.5351 | 270 176.5122 | 74 290 929 712 | 20 489 837 282 |
| 2 | 166 978.2675 | 135 088.2561 | 18 584 053 331 | 5 434 279 020 |
| 3 | 83 489.13377 | 67 544.12806 | 4 646 721 051 | 1 397 349 175 |
| 4 | 41 744.56688 | 33 772.06403 | 1 161 724 498 | 354 171 696.1 |
| 5 | 20 872.28344 | 16 886.03202 | 290 433 889.1 | 89 146 390.06 |
| 10 | 652.2588575 | 527.6885005 | 283 627.744 | 87 627.4025 |
| 15 | 20.3830893 | 16.49026564 | 276.9802196 | 85.59103372 |
| 20 | 0.636971541 | 0.515320801 | 0.270488496 | 0.083585525 |
| 30 | 0.000622043 | 0.000503243 | 2.57958E-07 | 7.97135E-08 |
| 40 | 6.07463E-07 | 4.91448E-07 | 2.46008E-13 | 7.61369E-14 |

| | | | | |
|---|---|---|---|---|
| … | …. | … | … | … |
| 135 | 1.53345E-35 | Side length less than the Planck length! | | |
| n | $\frac{2\pi R}{60 * 2^n}$ | $\frac{2\pi R * cos\left(\frac{\pi}{5}\right)}{60 * 2^n}$ | $\frac{\pi R^2 * sin\left(\frac{\pi}{45 * 2^n}\right)}{60 * 2^n}$ | [Appendix C] |

Table 2. Rectangle dimensions for QUTMS

***Trace***

For rectangles, there are 3 possible traces, after considering the rotations and reverses: an ‘n’ shape, an ‘x’ shape, and the ‘z’ shape. We have chosen the ‘z’ shape; interestingly, using this labelling produces a trace that is an iteration of the ‘Z-order’ curve (or Lebesgue curve), as shown in Figure 10 below:

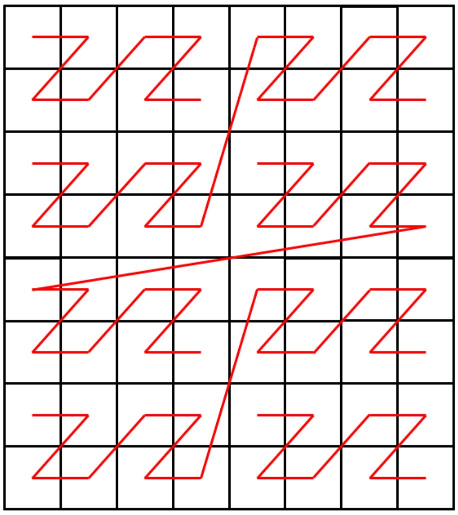


Figure 10. The trace within each level 0 rectangle

## 3. 3 Quaternary Rectangular System (QRS)

***Method***

Quaternary Rectangular System is a simplification of QUTMS. We define the level 0 grid by dividing latitude into 3 bands and longitude into 6 bands, resulting in eighteen 60 by 60 degree squares that can be subdivided with no exceptions. Figure 11 shows how this divides the globe at the first level of decomposition.

Figure 11. The Quaternary Rectangular System (QRS) Grid with one level of subdivision

The level 0 grid squares are numbered left to right, top to bottom, from 1 to 18, as seen in figure 12. A reference longitude boundary is fixed at 0 degrees and the latitude boundaries are fixed at +/- 30 degrees. A quaternary trail is established in the same way as QUTMS (using the same labelling) and this is encoded into base 32.

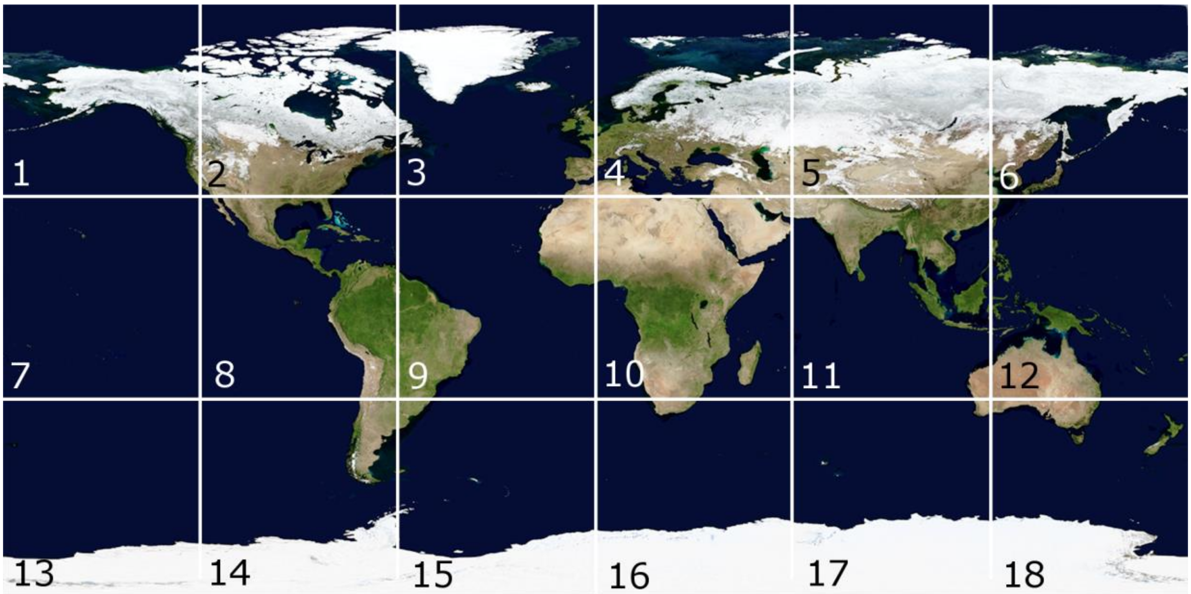



Figure 12. The Quaternary Rectangular System (QRS) level 0 grid squares

***Example***

Figure 13 shows the QRS grid over central London. The proximity and memorability are the same as QTS, which is expected due to the level 0 grids being similar in size (see Table 3). When compared to the QUTMS grid, the rectangles are not as tall; this is due to the divisions in QRS being 'squares' in terms of latitude and longitude.

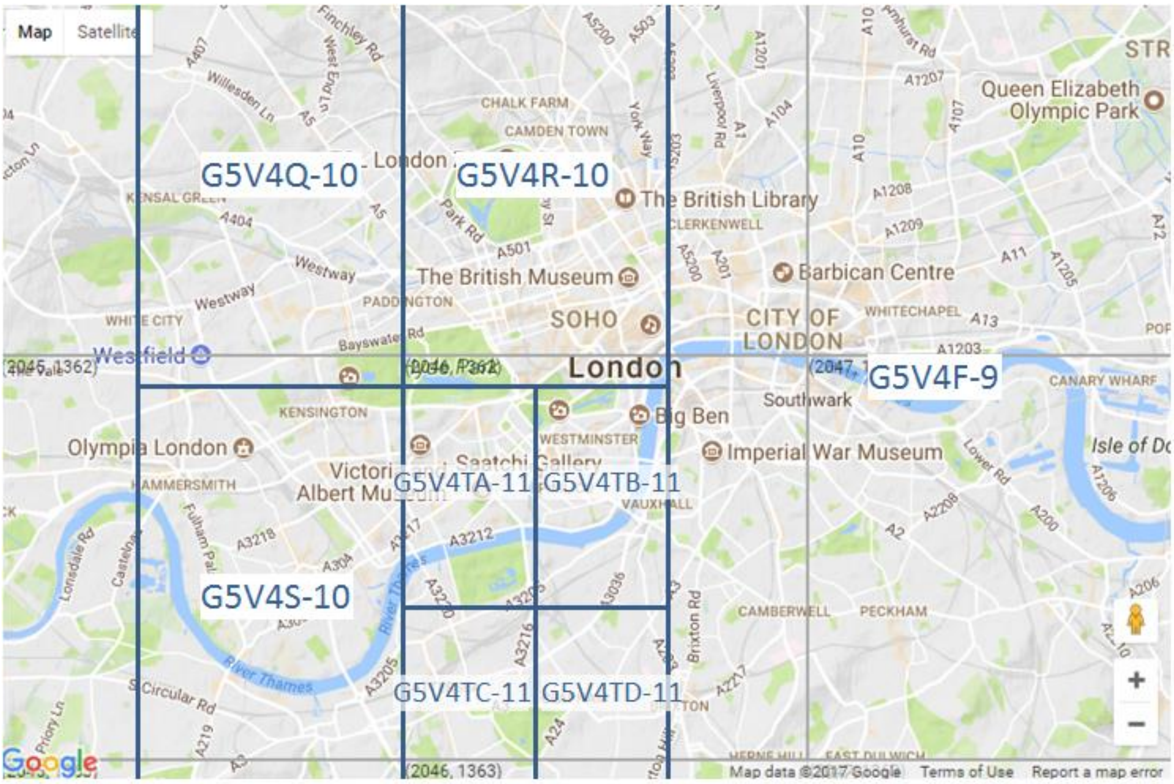



Figure 13. The QRS grid over central London
Note: Geognomo now requires "QRS:" entered before the alpha-numeric code in order to find a location.

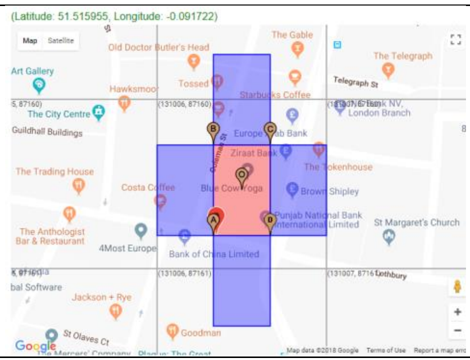
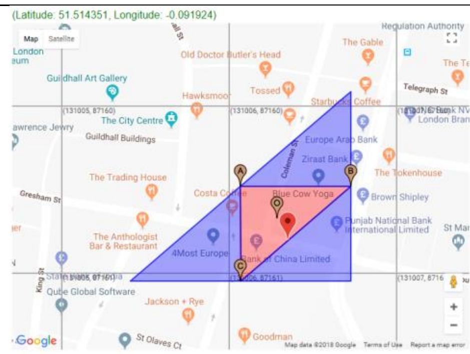
| Method | QRS | QTS |
|---|---|---|
| |  |  |
| GeoGnomo Code | QRS:**G5V4UWWP**-17 | QTS:**F49PUR6F**-17 |
| Neighbour A | QRS:**G5V4UW6F**-17 | QTS:**F49PUR6E**-17 |
| Neighbour B | QRS:**G5V4UWWN**-17 | QTS:**F49PUR8P**-17 |
| Neighbour C | QRS:**G5V4UWWO**-17 | QTS:**F49PUR9K**-17 |
| Neighbour D | QRS:**G5V4UWXK**-17 | N/A |

Table 3. QRS and QTS codes at level 17 for latitude: 51.514896 and longitude: -0.0901525

### *Grid details*

We use the same formula as for the QUTMS grid, updated to reflect the simplified divisions. The 'row X' dimensions are not shown but it is important to remember that the effect at high latitudes would be the same as seen in the QUTMS table.

| Level | Side length along equator (m) | Area at equator ($m^2$) |
|---|---|---|
| 1 | 6 679 130.701 | 1.50614E+13 |
| 2 | 3 339 565.351 | 4.07559E+12 |
| 3 | 1 669 782.675 | 1.03886E+12 |
| 4 | 834, 891.3377 | 2.60972E+11 |
| 5 | 417 445.6688 | 65 321 592 193 |
| 10 | 13 045.17715 | 6 3816 217.56 |
| 15 | 407.661786 | 62 320.54938 |
| 20 | 12.73943081 | 60.85991153 |
| 30 | 0.01244085 | 5.80405E-05 |
| 40 | 1.21493E-05 | 5.53518E-11 |
| … | …. | … |
| 140 | Side length less than the Planck length! | |
| n | $\frac{2\pi R}{60 * 2^n}$ | $\frac{\pi R^2 * sin\left(\frac{\pi}{45 * 2^n}\right)}{60 * 2^n}$ |

Table 4. Rectangle dimensions for QRS

***Trace***

Because the labelling for subdivisions is the same as QUTMS, the trace is also the same within each level 0 rectangle as shown in Figure 10.

## 3. 4 Variable Rectangular System (VRS)

***Method***

Variable Rectangular System uses a separate method from the quaternary trail method. Codes are generated from a rectangular area that may be specified through a 'click and drag' selection, and represent this selected area.

Once the area has been selected, the approach is to round the coordinate values so that the information can be stored in a code of memorable length, but the rounding is chosen so that the area remains a relatively close approximation to the original selection. Once rounded values have been chosen, they are organised in a predetermined way into a numerical code that stores only the decimal values needed to reproduce the rounded rectangle. This is then encoded into base 32 using a modified version of the method proposed by (Graham-Cumming 2006); this method is more suitable for the numerical code than the method we have used in the quaternary systems.

In particular, the selected area is defined by the latitude and longitude coordinates of the bottom-left and top-right corners. The latitude and longitude differences, i.e., the side lengths in degrees, are then rounded to 2 significant figures.  This ensures that the rounding error is restricted to a small percentage of the size of the rectangle.

The coordinates of the bottom-left corner are then rounded to the same number of decimal places as the rounded difference. The numerical digits of the corner position and the 2 digits of difference for latitude and longitude are joined together to form a numerical code, as shown in Figure 14.  This is done in such a way that proximity is more likely to be maintained.  The numerical code is, then, encoded in base 32 using the new method to produce the final code. This method is preferable as it does not add any digits to the code, resulting in the shortest possible codes that retain all the numerical information, which is necessary for keeping the codes generated by this method to a memorable length.

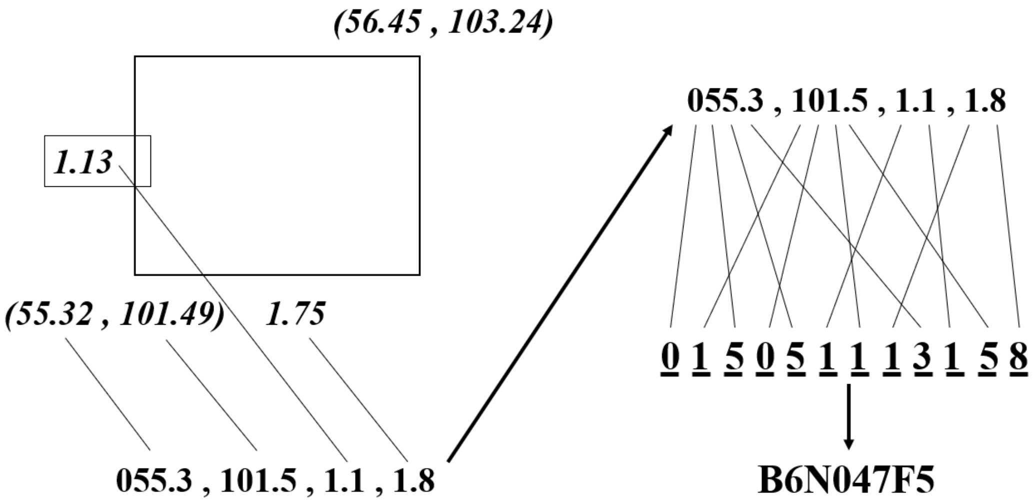



Figure 14. VRS code calculation

***Examples***

This example will illustrate the significant improvement in aggregation this system makes over the quaternary systems. One potential drawback of using any grid-based system is the chance that a single code does not describe a particular area very well; the area in question may cover the corners of several regions, thus overlapping them all but filling none of them to a significant extent. Figure 15 shows Hyde Park in London at (latitude 51.507273, longitude -0.165739) using QTS at level 13; the corresponding code is QTS:F49ON4A-13.

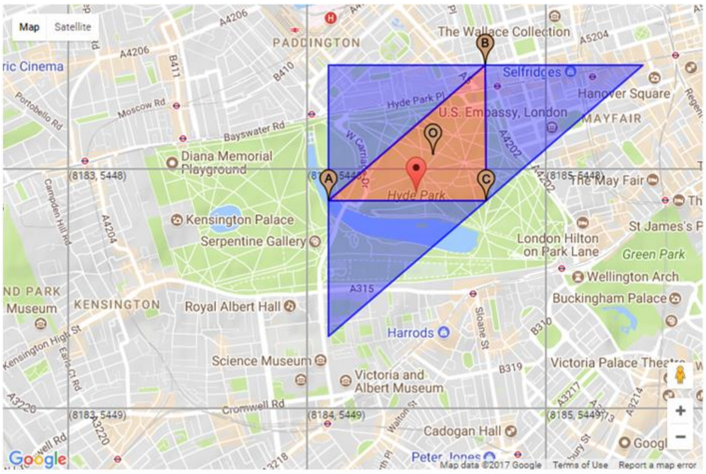



Figure 15. Hyde Park as seen using the QTS at level 13

Figure 16 below shows how Hyde Park can be selected much more accurately; the dashed line shows an initial selection and the shaded area shows the snapped area that generates the code.

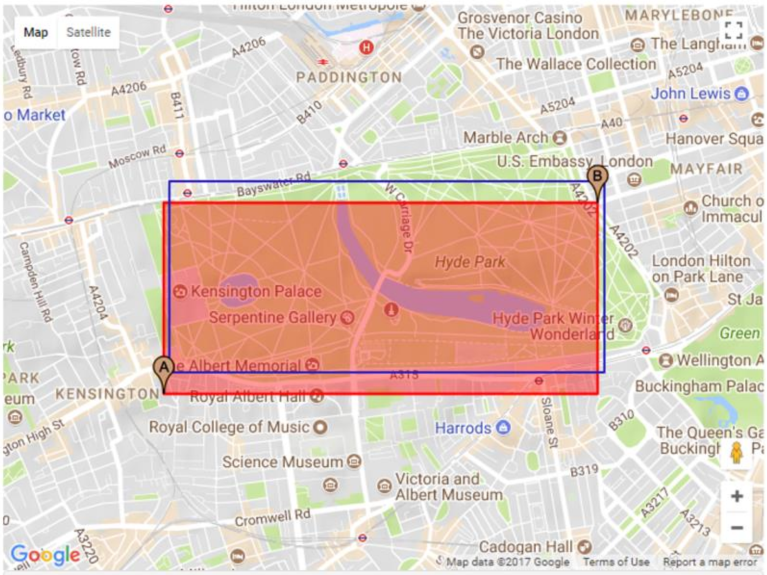



Figure 16. Hyde Park as seen using a VRS selection (shaded part is the coded area and the outlined rectangle is the initial dragged selection)

The VRS code is $VRS\colon RUYR3YV0AH$, which is only one character longer than the QTS code but greatly improves the control over the specified area. The snap distance is noticeable, but small relative to the size of the selection.

Figure 17 shows VRS codes over the same area of central London used in the earlier examples. It is important to remember that VRS is not fixed to this particular layout, whereas the other systems are fixed at each level; this example is constructed purely for comparison with the other examples.

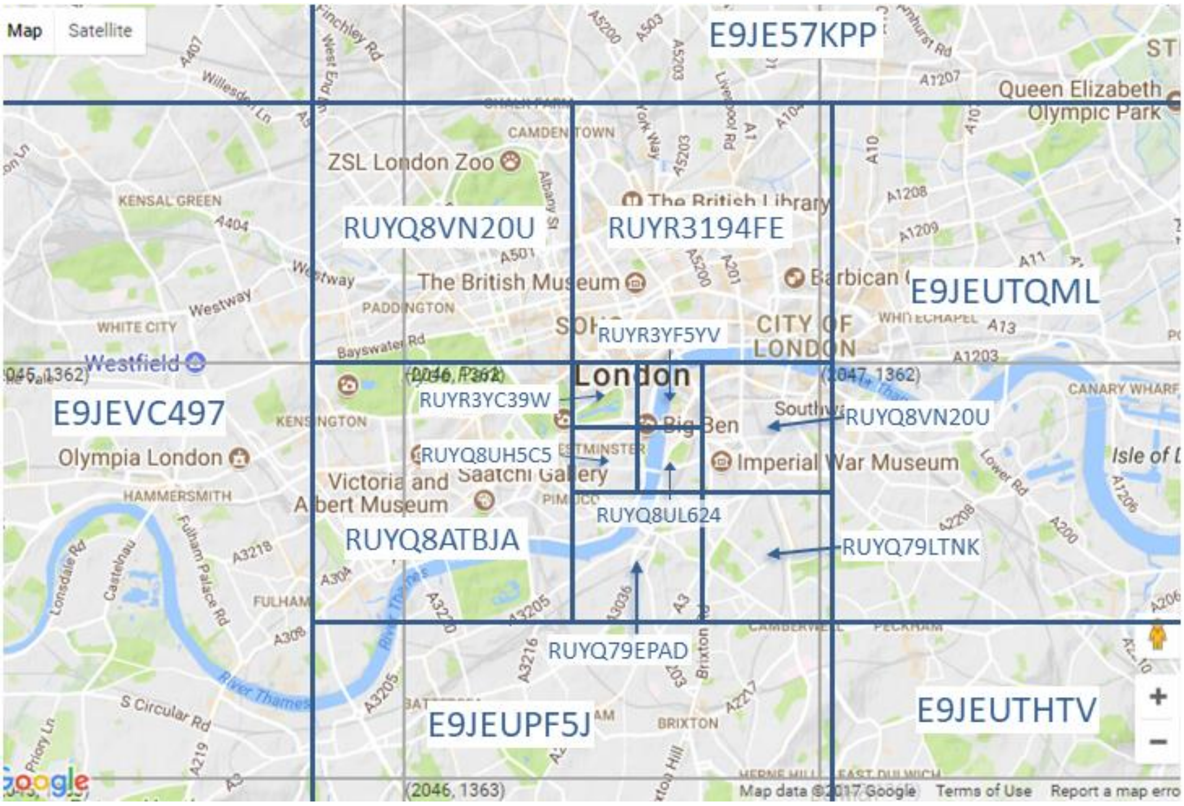



Figure 17. A grid of VRS selections over central London

Note: Geognomo now requires "VRS:" entered before the alphanumeric code in order to find a location.

While codes have a good proximity relationship at similar sizes, there is a cut-off where the rounding of coordinates changes and codes become unrelated.

Although VRS doesn't have levels, different lengths of the code denote different spatial scales, as demonstrated in Table 5. This is because the length of the code is, essentially, determined by how many decimal digits it requires to represent the box in latitude/longitude.

| Side length of box (degrees) | Length of code |
|---|---|
| 100+ | 4 |
| 10 - 100 | 6 |
| 1 - 10 | 8 |
| 1.1   1 | 9 |
| 0.01 - 0.1 | 10 |
| 0.001 – 0.01 | 12 |

Table 5. Relationship between spatial scale and length of code.

***Grid details***

Because the system does not have a level parameter or a fixed grid, these terms are not presented in the same way as in the previous sections. Instead, Table 6 summarises the 'error' (the difference between the area that the user selects and the area the code actually corresponds to) introduced by the rounding for each group of side lengths (determined by the number of decimal places rounding occurs at) in degrees. The maximum error is always 10% of the minimum side length; for example, a side stretching from 24.999999 degrees to 125 degrees (of length 100.000001 degrees) will be rounded to 20 degrees and 130 degrees, introducing 5 degrees of rounding error (or 5%) at each end.

| Side length (degrees) | Minimum side length along equator (m) | Maximum rounding error (m) |
|---|---|---|
| 100-360 | 1 113 1884.5 | 1 113 188.45 |
| 10-100 | 1 113 188.45 | 111 318.845 |
| 1-10 | 111 318.845 | 11 131.8845 |
| 0.1-1 | 11 131.8845 | 1 113.18845 |
| 0.01-0.1 | 1 113.18845 | 111.318845 |
| 0.001-0.01 | 111.318845 | 11.1318845 |
| 0.0001-0.001 | 11.1318845 | 1.11318845 |
| 0.00001-0.0001 | 1.11318845 | 0.111318845 |

Table 6. Side lengths and rounding errors of VRS selections

***Trace***

While the trace exists for VRS, it is much harder to visualise than the quaternary systems as the areas themselves overlap and vary in size. ,rendering it ineffective for this method. Furthermore, the list of possible codes itself is more complicated due to the rounding. For example, a 10x10 box will be encoded based on rounding implied by the 10 to 100 degree side length, but it is possible to construct a code that describes the same box as if it were rounded with 100 to 360 degree side lengths; this second code would not be included in the list of possible codes. These examples occur over all area sizes, and so explicitly identifying the list of possible codes would be more difficult than the quaternary systems.

***Refinement***

A refinement to this method would be to introduce a feature that accounts for the differences in longitudinal distance at latitudes near the poles compared to latitudes near the equator. As latitude moves away from the equator, the surface distance corresponding to each degree of longitude diminishes, until it reaches 0 at the poles. An approximate method to rectify this is to transform the rectangle on the map into a trapezium based on the latitude difference between the top and bottom edges. The edge nearest a pole would be longer, to account for the diminishing surface distance, and so the trapezium would approximately represent a real rectangle at the surface. Figure 18 is an approximate sketch of what this would look like in principle:

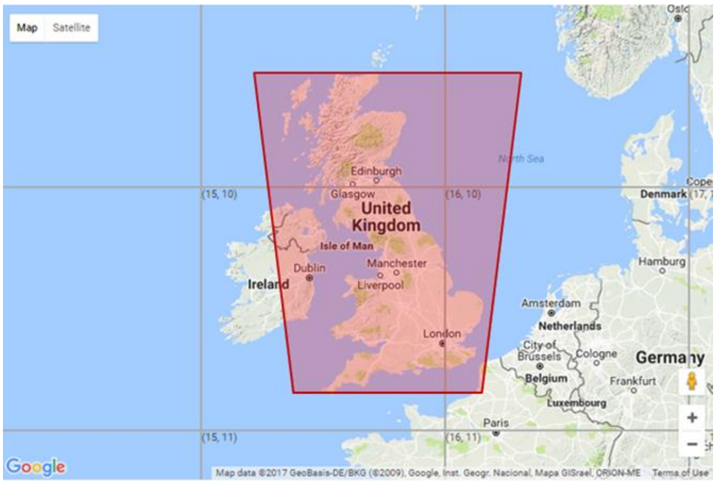



Figure 18. Projection mode over the UK

This method would only be approximate as the sides would be kept as straight edges, whereas the actual relationship is not linear. This implies that calculations are needed as the user selects the area to determine the angles of the trapezium. The advantage of adding this functionality would be most notable for large areas with wide latitude ranges, where a ‘coordinate rectangle’ differs greatly from a real rectangle.

## 4. **Comparison**

Table 7 summarises the relative performance of the four systems using the qualities put forward in the introduction. It includes measurable quantities such as the length of codes for an area of given size and our opinion on aspects such as aggregation, which would be very difficult to objectively measure over the entire globe. The table also includes a judgement of latitude/longitude point coordinates for comparison, as it was our goal to improve on this system.

***Memorability***

Memorability was judged mostly on the length of codes or coordinates. In order to specify an area, using latitude and longitude coordinates requires multiple points to be given and is, therefore, the least memorable, especially for small areas where high precision is needed for each point. VRS requires a couple more characters than the quaternary systems, which are very similar to each other. None are rated as ‘Very Good’ because codes are generally ‘random’ strings rather than recognisable English words or names.

***Aggregation***

Aggregation was judged on opinion and experimentation. Most applications of latitude and longitude for modelling and analysis partition the geographic coordinate grid into latitude-longitude blocks of varying degrees. For the quaternary systems, the ability to adjust the level allows them to describe areas of varying sizes, and even areas small enough to be treated as points given the accuracy of GPS systems. VRS is the best in our opinion due to areas not being fixed to a grid, allowing areas to be different shapes and sizes and be moved more precisely by the user. However, for actual global applications, VRS could be more complicated and more difficult to compare levels of aggregation than with the quaternary systems. It is easy to find examples where it fails along with the other systems; one being any case where an area crosses over the 180 degree longitude line.

***Proximity***

Proximity was judged through experimentation and using the examples over central London. As well as this, the ‘worst case’ instances were considered. For example, there are 2 examples where latitude and longitude coordinates fail to represent close proximity: 2 points that are close to, but on different sides of the 180 degree longitude line and 2 points close to the poles that happen to have opposite longitudes. In each case the longitude coordinates would suggest that points are very far apart when the reality is the opposite. The quaternary systems were judged to be ‘Good’ because they largely give similar codes when areas are close together and this is maintained over all levels. However, the level 0 grid introduces large differences in neighbouring codes on the boundaries. In VRS, the scaling determines the length of the code, so the length of the code is an indicator of the scaling. While the scaling can lead to different codes for very similar areas, usually this does not happen unless the areas are of very different sizes (such that the length of the longer side in degrees has a different exponent in standard form, for instance 5 degrees and 50 degrees). If the scaling falls within the same code size, the proximity is very good and there aren't, really, any 'sudden' jumps (like when neighbouring triangles are in different level 0 triangles in QTS). Because of the fact that at a fixed scale, it could be, even, said to have better proximity than the quaternary systems, VRS has been assigned ‘Good’ proximity.

***Scale***

Again, latitude and longitude coordinates are judged to be ‘Poor’ due to being limited to points only. The quaternary systems are judged to be ‘Good’ because a wide range of sizes are covered, and the options are fairly dense throughout the range, although for larger sizes the options are sparse. VRS is a further improvement, with the range being greater and options being very dense throughout the range, especially at larger sizes.

| System | Memorability | Aggregation | Proximity | Scale |
|---|---|---|---|---|
| **Lat/long coordinates** | Poor | Average | Very Good | Poor |
| **QTS** | Good | Average | Good | Good |
| **QUTMS** | Good | Average | Good | Good |
| **QRS** | Good | Average | Good | Good |
| **VRS** | Average | Good | Good | Very Good |

Table 7. Rating of the four systems and latitude/longitude coordinates

In order to provide an aggregate assessment, a numerical rating of each of the four qualities (where 1=Poor, 2=Average, 3=Good, and 4=Very Good) provides the following summary.

| System | Memorability | Aggregation | Proximity | Scale | Rating |
|---|---|---|---|---|---|
| **Lat/long coordinates** | 1 | 2 | 4 | 1 | 2 |
| **QTS** | 3 | 2 | 3 | 3 | 2.75 |
| **QUTMS** | 3 | 2 | 3 | 3 | 2.75 |
| **QRS** | 3 | 2 | 3 | 3 | 2.75 |
| **VRS** | 2 | 3 | 3 | 4 | 3 |

Table 8. Aggregate assessment of the four systems and latitude/longitude coordinates

The above assumes that the four factors have equal weight. This might be unrealistic in many cases. For example, if cargo firm needs to distribute supplies to a specific area in Africa, scalability might be more important than memorability.

***Comparison with other systems***

What3words (https://what3words.com/) divides the globe in a grid of equal 3x3 meter squares. Each square has been assigned a 3 word chain, for example, daring.lion.race. For English speakers, the memorability of the 'code' is significantly higher than other methods, and what3words has been adding other languages.

Geohash (https://www.movable-type.co.uk/scripts/geohash.html) has more similarities with the three quaternary systems. Geohash initially divides the globe into 32 rectangles of different height, depending on their distance from the equator. Each rectangle is then subdivided into 32 rectangles, and so on and so forth. Like GeoGnomo, it uses alphanumeric strings, but the number of characters specifies the level of subdivision. Although proximity, is good in general, there are neighbouring areas that fall under different level 0 rectangles, which have very different codes.

| System | Memorability | Aggregation | Proximity | Scale |
|---|---|---|---|---|
| **What3words** | Very Good | Average | Poor | Poor |
| **Geohash** | Good | Average | Good | Good |

Table 9. Ratings for What3words and Geohash

## 5. Further Ideas

### *Hexagonal Systems*

Across the four systems in GeoGnomo, only the possibilities of triangles and rectangles have been explored. Hexagons are a popular choice for area division (and common in nature), especially on a 2-dimensional plane, due to their mathematical properties (White & Kiester, 2008). Hexagons would be more complicated than current systems in GeoGnomo; hexagonal tiling of a sphere could also be based on an icosahedron (or octahedron or tetrahedron), but because hexagons do not subdivide into further hexagons, a new ‘level’ system would need to be devised. Furthermore, hexagons would overlap the edges of the icosahedral triangles (see White et al. 1998, figure 2). There would, also, need to be separate equations for 12 pentagons to be introduced at each vertex of the icosahedron. Aside from the complexity, a hexagonal system would not have significant disadvantages when compared to the other systems. Like QTS, the hexagon areas would have small variation over the whole globe. Furthermore, it is possible to specify hexagonal hierarchies, so that the position of the corners are never fixed as the level varies (Sahr K., 2011), avoiding cases such as the one seen in Figure 14.

### *Altitude*

Next generation georeferencing applications will include 3-dimensional virtual globes that will allow a broad spectrum of users, including scientists, businesses and individuals, to interactively visualize, analyse, model, manipulate, and generate geospatial big data (Sahr 2013). For example, they will allow insurers to do a broader range of analysis with regards to how their business relates to a location in 3-dimensional view. Is a building better modelled as a set of blocks in 3 dimensions rather than a planar map of portions of the Earth’s surface?

In our study, we attempt to block the third dimension – height/altitude based on the average equatorial side lengths, as well as using an explicit measurement in metres. The terms we use to describe the third dimension can be one of the following:

- MA (meters absolute from centre of reference ellipsoid)
- MS (meters from sea level)
- GA (GeoGnomo units from centre of reference ellipsoid)
- GS (GeoGnomo units from sea level)

For example, if we were to add altitude to QTS using Table 1, an altitude at 15MS is equivalent to 1GS at level 19 and will change to 2GS at level 20. Big Ben is 96m tall, starting at 19m above sea level and so we can describe the top of Big Ben with a 3 dimensional QTS code at level 19:

$QTS: F49PWP23A@7GS - 19$or $Foxtrot, 4,9, Papa, Whiskey, Papa, 2,3, Alpha@7GS - 19$

Or alternatively, using metres from centre of geoid as another example:

$QTS: F49PWP23A@96MA - 19$or

$Foxtrot, 4,9, Papa, Whiskey, Papa, 2,3, Alpha@96MA - 19$

***Path Tracing***

A potential addition to the quaternary systems would be a feature that uses the neighbours to encode a path, or more generally a list of georeferences that specify an irregular area. The encoded list would have to be given in addition to the first code, so the list part would have to be encoded efficiently to avoid making the overall georeference too long. To specify an area using only direct neighbours, it would be necessary to minimise backtracking in the cases where a continuous path does not exist. A possible approach to this problem would be to model the grid's underlying graph structure and consider the travelling salesman problem.

## Conclusion

This paper presents four georeferencing systems, each using a different strategy in an attempt to improve on latitude and longitude coordinates as a method for geostamping. The main strength of longitude and latitude coordinates is that by numerically representing point locations, they have excellent proximity. The ability to define areas of varying size, however, allows for better aggregation. Using a base 32 system allows codes of memorable length to be produced. The three quaternary systems are closely related in design and largely improve on memorability, aggregation, and scale. VRS was designed to further improve aggregation, which came at a small cost in memorability. Experimenting with other shapes, such as the hexagon, might produce other grid systems that fall in between VRS and the quaternary systems. Of the three quaternary systems, the simplicity of QRS makes it stronger over most of the globe, with the exception being at high latitudes when QTS is the strongest. Overall, the VRS is the strongest due to the control and flexibility. We have suggested hexagonal systems, altitude coding, and path tracing as areas for further research.

## References


Dutton, G. 1989. Planetary modelling via hierarchical tessellation. Proceedings of the Eleventh International Conference on Computer-Assisted Cartography (Auto-Carto 9). Anderson E, ed. American Congress on Surveying and Mapping, Baltimore, MD. pp 462-471.

Dutton, G. 1990. Locational properties of quaternary triangular meshes. Proceedings of the 4th International Symposium on Spatial Data Handling. Department of Geography, University of Zurich. pp 901-910.

Egnor, R., Miller, C. & Hauser, M.D. 2004. "Nonhuman Primate Communication" (PDF). Encyclopedia of Language and Linguistics (2nd ed.). Elsevier.

Goodchild, M.F., Shiren, Y., Dutton, G. 1991. Spatial data representation and basic operations on triangular hierarchical data structure. Technical Paper 91-8. National Center for Geographical Information and Analysis, University of California, Santa Barbara, CA. 39 pp.

Goodchild, M. F., & Shiren, Y. 1992. A Hierarchical Spatial Data Structure for Global Geographic Information Systems. CVGIP: Graphical Models and Image Processing. 54, pp. 31-44. Elsevier Inc.

Graham-Cumming, J. 2006. A Simple Code for Entering Latitude and Longitude to GPS Devices. Retrieved June 2017, from http://blog.jgc.org/2006/07/simple-code-for-entering-latitude-and.html

Hauser, M.D., Chomsky, N., Fitch, W.T. 2002. The faculty of language: what is it, who has it, and how did it evolve? Science 298(5598):1569-1579.

Lee, M., & Samet, H. 1998. Traversing the Triangle Elements of an Icosahedral Spherical Representation in Constant-Time. *Proceedings 8th International Symposium on Spatial Data Handling*, (pp. 22-23). Vancouver, Canada.

Lukatela, H. 1987. Hipparchus geopositioning model: an overview. Proceedings of the Eighth International Conference on Computer-Assisted Cartography (Auto-Carto 8). American Congress on Surveying and Mapping. pp 87-96.

Mainelli, M., & Manson, B. 2016. Chain Reaction: How Blockchain Technology Might Transform Wholesale Insurance. *Long Finance and PwC*.

Mainelli, M., & Manson, B. 2017. From Slips To Smart Contracts: Intelligent Technology In The London Wholesale Insurance Market. *Long Finance and London Market Group*.

Mainelli, M., & Smith, M. 2015. Sharing Ledgers For Sharing Economies: An Exploration Of Mutual Distributed Ledgers (aka blockchain technology). *Journal of Financial Perspectives*, *3*(3).

Riley, J.R., Greggers, U., Smith, A.D., Reynolds, D.R. Menzel, R. 2005. The flight paths of honeybees recruited by the waggle dance. Nature 435:205–207.

Sadourny R, Arakawa A, Mintz Y. 1968. Integration of the nondivergent barotropic vorticity equation with an icosahedral-hexagonal grid for the sphere. Monthly Weather Review 96(6):351-356.

Sahr, K. 2013. On the Optimal Representation of Vector Location Using Fixed Width. *International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences, Volume XL-4/W2.* Xuzhou, Jiangsu, China.

Sahr, K. 2011. Hexagonal Discrete Global Grid Systems for Geospatial Computing. Archives of Photogrammetry, Cartography and Remote Sensing, 22, 363-376.

Swinbank R, Purser R.J. 2006. Fibonacci grids: a novel approach to global modeling. Quarterly Journal of the royal Meteorological Society 132:1769-1793.

Tobler, W., Chen, Z-T. 1986. A quadtree for global information storage. Geographical Analysis 18(4):360-371.

Tong, X., Ben, J., Wang, Y. 2010. A new high effective hexagonal discrete global grid system: hexagonal quad balanced structure. Proceedings of the 18$^{th}$ International Conference on Geoinformatics. 6 pp.

White, D., Kimerling, A. J., Sahr, K., & Song, L. 1998. Comparing area and shape distortion on polyhedral-based recursive. *International Journal of Geographical Information Science, 12*(8), 805 -82

White, D., & Kiester, R. 2008. Topology Matters: Network topology affects outcomes from community ecology neutral models. Computers, Environment and Urban Systems, 30(2), 165-171.

Williamson, D.L. 1968. Integration of the barotropic vorticity equation on a spherical geodesic grid. Tellus, 20(4), 642–653.

## Appendix A

### *Method for QTS*

Figure A1 shows the net of the icosahedron. For calculations we use radians and adjust the ranges so that $-\frac{\pi}{2} < \phi < \frac{\pi}{2}$ and $0 < \lambda < 2\pi$. We set $\lambda = 0$ to be the left edge of triangle 1. The 10 vertices that are not at the poles are located at $\phi = \pm arctan\left(\frac{1}{2}\right)$ and are spaced $\frac{2\pi}{5}$ longitude apart, although the bottom 5 are offset by $\frac{+\pi}{5}$. For a given $(\phi, \lambda)$, we first decide whether this defines a point in the top pyramid, the bottom pyramid or the middle band of the icosahedron by its latitude. If the level 0 triangle is in either the top or bottom row, the edges of the triangles represent a single longitude value, so we can find $a_0$ by:

Top pyramid: $$a_0 = \lfloor \frac{5\lambda}{2\pi} \rfloor + 1 \qquad \left(\phi \geq \arctan(0.5)\right) \qquad (1)$$

Bottom pyramid: $$a_0 = \lfloor \frac{5\left(\lambda - \frac{\pi}{5}\right)}{2\pi} \rfloor + 15 + f(\lambda) \qquad \left(\phi \leq -\arctan(0.5)\right) \qquad (2)$$

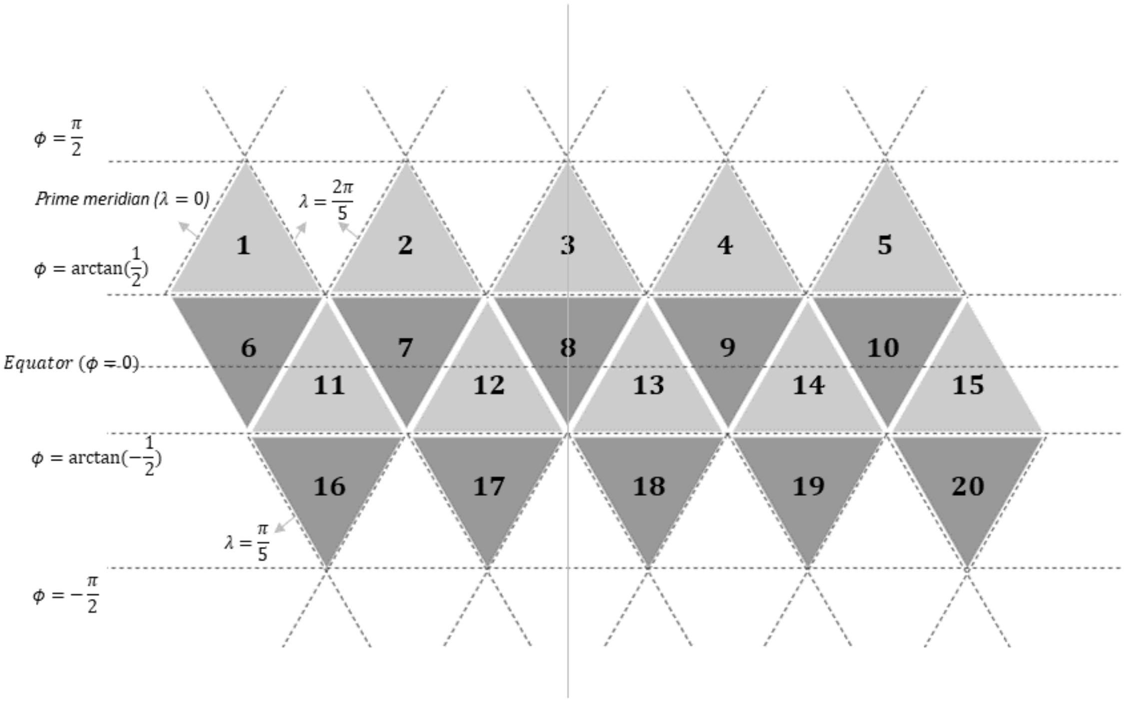



Figure A1. The icosahedron net

If it is in the middle band, we treat the (*x*,*y*)-plane as a linear scaling of $(\phi, \lambda)$. We use equation (3) to find which of the top pyramids the point is located south of, and then use equations (4) and (5) to find the two edges within the middle band:

$$m = \lfloor \frac{5\lambda}{2\pi} \rfloor + 1 \quad (3)$$

Left edges: $$\phi_L = \frac{-2arctan\left(\frac{1}{2}\right)}{\frac{\pi}{5}}\lambda + (4m-3)arctan\left(\frac{1}{2}\right) \quad (4)$$

Right edges: $$\phi_R = \frac{2arctan\left(\frac{1}{2}\right)}{\frac{\pi}{5}}\lambda - (4m-1)arctan\left(\frac{1}{2}\right) \quad (5)$$

We then test the coordinates against the edges to find out which triangle the point lies in:

$$a_0 = \lfloor \frac{5\lambda}{2\pi} \rfloor + 10 + g(\lambda) \qquad if\ \phi < \phi_L$$

$$a_0 = \lfloor \frac{5\lambda}{2\pi} \rfloor + 11 \qquad if\ \phi \leq \phi_R \quad (6)$$

$$a_0 = \lfloor \frac{5\lambda}{2\pi} \rfloor + 6 \qquad if\ \phi \geq \phi_L\ and\ \phi > \phi_R$$

Our method then follows (Goodchild and Shiren 1992); within each level 0 triangle, we define new (*x*,*y*) coordinates for the point and then transform these to ‘triangle address coordinates’ based on the subdivisions, allowing us to determine the quaternary code. Once $a_0$ is known, we set the edge length of the smallest decomposed triangles to 1 and so the triangle itself has vertices at (0, 0), ($2^n$, 0) and ($2^{n-1}, 2^{n-1}\sqrt{3}$), as shown in Figure A2. The latitude/longitude coordinates are adjusted relative to the bottom left corner of the level 0 triangle. The edges of the triangle are described by:

Left edge: $y = \sqrt{3}x$ or $\lambda = 0$ (7)

Right edge: $y = (2^{n-1} - x)\sqrt{3}$ or $\lambda = \frac{2\pi}{5}$ (8)

Bottom edge: $y = 0$ or $\phi = 0$ (9)

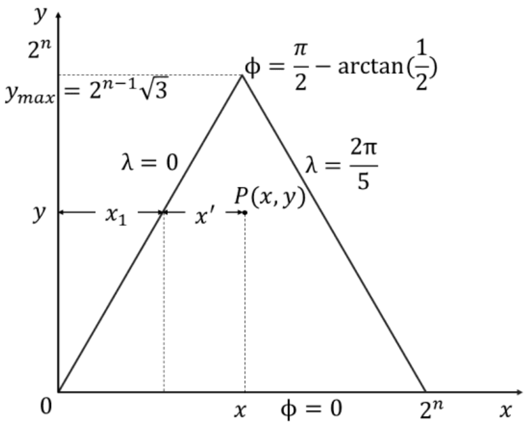



Figure A2. Relationship between Geographic Coordinates and Cartesian Coordinates

The relation between $y$ and $\phi$ is therefore

$$y = \frac{2^{n-1}\sqrt{3}}{\frac{\pi}{2} - arctan\left(\frac{1}{2}\right)}\phi \tag{10}$$

and the relation between $x$ and $\lambda, \phi$

$$x = x' + x_1 = 2^{n-1}\left[\frac{\phi}{\frac{\pi}{2} - arctan\left(\frac{1}{2}\right)} + \frac{5\lambda}{\pi}\left(1 - \frac{\phi}{\frac{\pi}{2} - arctan\left(\frac{1}{2}\right)}\right)\right] \tag{11}$$

and the expressions for transformation of $x$ and $y$ to longitude and latitude in the triangle are

$$r = \frac{\phi}{\frac{\pi}{2} - arctan\left(\frac{1}{2}\right)} \tag{12}$$

$$\lambda = \frac{\left(\frac{x}{2^{n-1}} - r\right)\pi}{5(1 - r)}, \phi = \frac{\frac{\pi}{2} - arctan\left(\frac{1}{2}\right)}{2^{n-1}\sqrt{3}}y \tag{13}$$

As for the middle band, we have already assumed that the edges are lines on a plane and so both x and y are assumed to be a scaling of longitude and latitude. Therefore, we have

$$x = \frac{5.\,2^{n-1}}{\pi}\lambda \tag{14}$$

and

$$y = \frac{2^{n-2}\sqrt{3}}{arctan\left(\frac{1}{2}\right)}\phi \tag{15}$$

It follows that the expressions for transformation of *x* and *y* to longitude and latitude are

$$\lambda = \frac{x\pi}{5.\,2^{n-1}}, \phi = \frac{arctan\left(\frac{1}{2}\right)}{2^{n-2}\sqrt{3}}y \tag{16}$$

When converting from (*x*,*y*) to ($\lambda$,$\phi$), the adjustments that would have been made according to Figure A1 are reversed following equations (13) and (16).

For conversion of Cartesian coordinates to triangle address coordinates, we first find the maximum number of rows *I* and columns *J* at the *n*-th level using:

$$I = 2^n, \qquad J = 2 * I - 1 \tag{17}$$

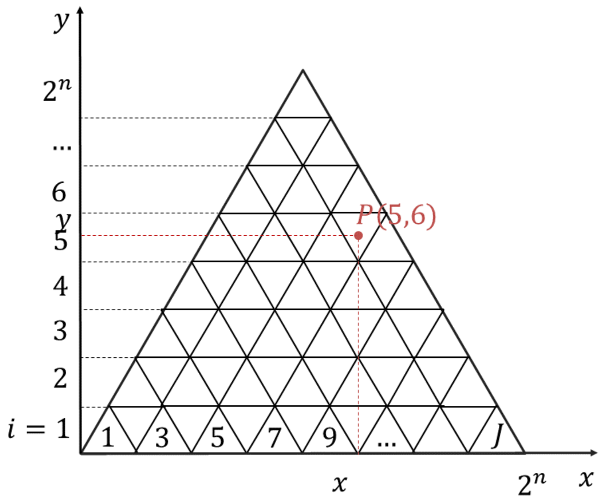



Figure A3. Relationship between Cartesian Coordinates and Triangle Address Coordinates (n=3)

From Figure A2 and Figure A3, since $y_{max}$ on the Cartesian plane corresponds to the maximum number of rows *I*, we have the following expression for the relation between *y* and *i:*

$$i = \left\lfloor\frac{y * I}{2^{n-1}\sqrt{3}}\right\rfloor + 1 \tag{18}$$

Each additional row offsets the triangles by 1/2 a unit in the *x* direction. Then initial column position $j'$ can found as:

$$j' = 2\left\lfloor x - \frac{i-1}{2}\right\rfloor + 1 \tag{19}$$

Note that $j'$ only takes odd values at this stage. To find out *j*, we need to find which side of the line the point lies on. From Figure A3, we have two general expressions to describe all the left and right edges of triangles:

Left edge: $$y_L = \sqrt{3}x + \left(\frac{1-j'}{2}\right)\sqrt{3} \tag{20}$$

Right edge: $$y_R = -\sqrt{3}x + \left(i + \frac{j'-1}{2}\right)\sqrt{3} \tag{21}$$

Then

$$\begin{aligned} j &= j' - 1 && if\ y > y_L \\ j &= j' + 1 && if\ y > y_R \\ j &= j' && if\ y \le y_L\ and\ y \le y_R \end{aligned} \tag{22}$$

The method for converting triangle address coordinates *(i,j)* into a quaternary code (or quaternary trail) $a_1 a_2 a_3 \ldots a_k$ takes *n* and *(i,j)* as inputs and then follows the zooming in path as described earlier.

From Equation (15), we have $I = 2^n$.

We start from $k = 1$.

$if\ \ i > \frac{I}{2}$ :

$a_k = 1\ ,\qquad I = \frac{I}{2}\ ,\qquad i = i - I\ ;$

*else*:

$if\ j < 2\left(\frac{I}{2} - i + 1\right)$ :

$a_k = 2\ ,\quad I = \frac{I}{2}\ ;$

*else* :

$if\ \ j > n$ :

$a_k = 3,\qquad I = \frac{I}{2}\ ,\qquad j = j - 2 * I;$

*else* :

$a_k = 0,\qquad j = j - n + 2 * i - 1,\qquad i = \frac{I}{2} - i + 1\ ;$

$k = k + 1\ ;$

Repeat these until k=n.

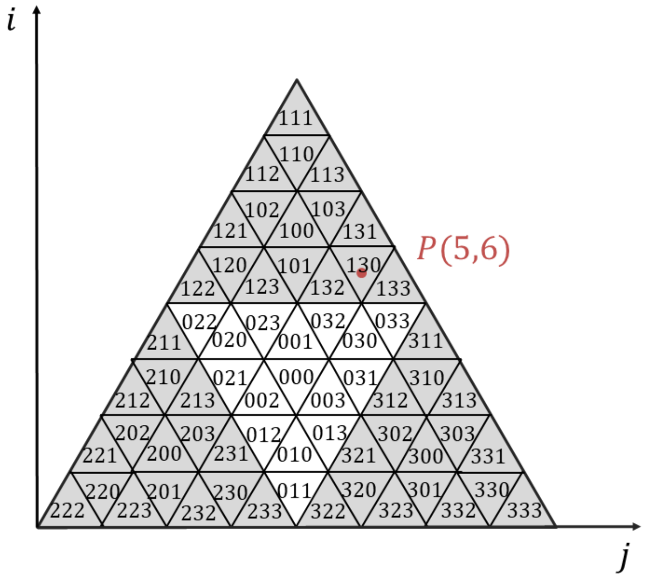



Figure A4. Quaternary code of triangles at level 3

To find the triangle address coordinates *(i, j)* at level *n* from the quaternary code $a_1 a_2 a_3 \ldots a_k$, the algorithm follows the path described by the quaternary code:

From Equation (13), we have $I = 2^n$. Set an initial orientation flag $P = 1$.

Starting from $k = 1$, $i = 1$, $j = 1$,

*if* $P = 1$:

*if* $a_k = 0$:

$$j = j + I - 1\,, \quad I = \frac{I}{2}\,, \quad P = -P\,;$$

*if* $a_k = 1$:

$$i = i + \frac{I}{2}\,, \quad I = \frac{I}{2}\,;$$

*if* $a_k = 2$:

$$I = \frac{I}{2}\,;$$

*if* $a_k = 3$:

$$j = j + I\,, \quad I = \frac{I}{2}\,;$$

*if* $P = -1$:

*if* $a_k = 0$:

$$i = i + \frac{I}{2}\,, \quad j = j - I + 1\,, \quad I = \frac{I}{2}\,, \quad P = -P\,;$$

*if* $a_k = 1$:

$$I = \frac{I}{2}\,;$$

*if* $a_k = 2$:

$$i = i + \frac{I}{2}\,, \quad j = j - I\,, \quad I = \frac{I}{2}\,;$$

*if* $a_k = 3$:

$$i = i + \frac{I}{2}\,, \quad I = \frac{I}{2}\,;$$

$k = k + 1$ ;

Repeat until k=n.

## Appendix B

### *Methods for QUTMS and QRS*

Using rectangles greatly reduces the need to manipulate latitude and longitude values before the quaternary trail can be established. In both cases $a_0$ is established by comparing the latitude and longitude values of the given point to the known coordinates of the grid. Once this is known, the latitude and longitude coordinates are adjusted to be relative to the bottom left corner. Instead of setting the side lengths to be dependent on $n$, we leave them in degrees according to the level 0 grid and label these lengths ‘latlength’ and ‘lonlength’. With the adjusted coordinates $(\phi', \lambda')$, the algorithm to establish the quaternary trail is as follows:

$for\ k = 1\ to\ n$:

$if\ \lambda' < lonlength/2$:

$if\ \phi' < latlength/2$:

$a_k = 2$

$if\ \phi' \geq latlength/2$:

$a_k = 0, \phi' = \phi' - latlength/2$

$if\ \lambda' \geq lonlength/2$:

$if\ \phi' < latlength/2$:

$a_k = 3, \lambda' = \lambda' - lonlength/2$

$if\ \phi' \geq latlength/2$:

$a_k = 1, \lambda' = \lambda' - lonlength/2, \phi' = \phi' - latlength/2$

## Appendix C

### *Methods for obtaining grid details and neighbours*

The orientation can be found to be upward or downward from $a_0$, *i* and *j* by:

$$P = (-1)^{\left(\lfloor\frac{a_0-1}{5}\rfloor mod2\right)+(jmod2)+1} \tag{23}$$

where $P = 1$ means the triangle is orientated upwards and $P = -1$ means the triangle is orientated downwards.

The Cartesian coordinates of the centroid and vertices of a triangle with given triangle address *(i, j)* are:

Centroid: $$(X_0, Y_0) = \left(\frac{i}{2} + \frac{j-1}{2}, \sqrt{3}\frac{(i-1+j\ mod\ 2)}{2} + \frac{(-1)^j}{\sqrt{3}}\right) \tag{24}$$

Left vertex: $$(X_{left}, Y_{left}) = \left(\frac{i-1}{2} + \frac{j-1}{2}, \frac{i-(j\ mod\ 2)}{2}\sqrt{3}\right) \tag{25}$$

Right vertex: $$(X_{right}, Y_{right}) = \left(\frac{i}{2} + \frac{j}{2}, \frac{i-(j\ mod\ 2)}{2}\sqrt{3}\right) \tag{26}$$

Top or Bottom vertex: $$(X_{top}, Y_{top}) = \left(\frac{i}{2} + \frac{j-1}{2}, \frac{i-1+(j\ mod\ 2)}{2}\sqrt{3}\right) \tag{27}$$

The area of a triangle can be calculated as follows: Starting with the top and bottom pyramids, the Earth surface area A between latitude $\phi_1$ and $\phi_2$covered by a level 0 triangle is

$$A = \frac{2\pi R^2}{5}(\sin\phi_2 - \sin\phi_1) \tag{28}$$

where R is the radius of the Earth. Given the triangle address *(i, j)*, the total number of triangles in the belt between $\phi$ and$\phi + \frac{\frac{\pi}{2} - arctan\left(\frac{1}{2}\right)}{I}$ is

$$N_\phi = 2(I - i + 1) - 1 \tag{29}$$

and the Earth surface area of a triangle is

$$\Delta A_{\phi} = \frac{A}{N_{\phi}} = \frac{\frac{2\pi R^2}{5}\left(sin\left(\phi + \frac{\frac{\pi}{2} - arctan\left(\frac{1}{2}\right)}{I}\right) - sin\phi\right)}{2(I - i + 1) - 1} \tag{30}$$

Similarly, for the middle bands, the Earth surface area A between latitude $\phi_1$ and $\phi_2$ covered by a level-0 triangle in the middle bands

$$A = \frac{\pi R^2}{5}(\sin\phi_2 - \sin\phi_1) \tag{31}$$

Given the triangle address *(i, j)*, the total number of triangles in the belt between $\phi$ and$\phi + \frac{2arctan\left(\frac{1}{2}\right)}{I}$ is

$$N_{\phi} = 2I - 1 \tag{32}$$

and the Earth surface area of a triangle is

$$\Delta A_{\phi} = \frac{A}{N_{\phi}} = \frac{\frac{\pi R^2}{5}\left(sin\left(\phi + \frac{2arctan\left(\frac{1}{2}\right)}{I}\right) - sin\phi\right)}{2I - 1} \tag{33}$$

Different algorithms for finding neighbours have been described by (Goodchild and Shiren 1992), (Lee and Samet 1998) and others. The algorithm for finding neighbours is much simpler in our schema with given triangle address coordinates *(i, j)*, level-0 digit $a_0$ and maximum decomposition level *n*.

First, the expressions to find the neighbours of initial level-0 triangles $a_0$ are:

North triangle:
$$\mathrm{north}(\mathrm{a}_0) = ((a_0 \ mod \ 10) > 4) * (a_0 - 5) \tag{34}$$

South triangle:
$$\mathrm{south}(\mathrm{a}_0) = ((a_0 \ mod \ 10) < 5) * (a_0 + 5) \tag{35}$$

East triangle:
$$\begin{aligned}\mathrm{east}(\mathrm{a}_0) = a_0 + 1 - 5 * (a_0 > 4) + 9 * (a_0 > 5) - 9 * (a_0 > 10)\\ -5 * (a_0 > 14) + 10 * (a_0 > 15) - 5 * (a_0 > 19)\end{aligned} \tag{36}$$

West triangle:
$$\begin{aligned}\mathrm{west}(\mathrm{a}_0) = a_0 + 4 - 5 * (a_0 > 1) + 10 * (a_0 > 5) - 5 * (a_0 > 6)\\ -9 * (a_0 > 10) + 9 * (a_0 > 15) - 5 * (a_0 > 16)\end{aligned} \tag{37}$$

where (A>B) is a logical test, equal to 1 if A>B is true or 0 if A>B is false. In a programming environment this is neater than having different equations for several cases.

Then, we denote the triangle *(i, j)* on face $\mathrm{a}_0$ as $(\mathrm{a}_0, i, j)$ and its three direct neighbours as left $(\mathrm{a}_{0\_\mathrm{left}}, i_{left}, j_{left})$, right $(\mathrm{a}_{0\_\mathrm{right}}, i_{right}, j_{right})$ and top $(\mathrm{a}_{0\_\mathrm{top}}, i_{top}, j_{top})$ at level *n*. The direct neighbours can be found by the following algorithm (pseudo-code):

(1) To find the left neighbour,

$if\ j = 1$:

$if\ 6 \le a_0 \le 15$: $\quad i_{left} = 2^n + 1 - i$ ;

*else:* $\quad i_{left} = i$ ;

$j_{left} = 2 * (2^n - i) + 1$;

$\mathrm{a}_{0\_\mathrm{left}} = \mathrm{west}(\mathrm{a}_0)$ ;

*else*:

$i_{left} = i$ ;

$j_{left} = j - 1$;

$\mathrm{a}_{0\_\mathrm{left}} = \mathrm{a}_0$ ;

(2) To find the right neighbour,

$if\ j = 2 * (2^n - i) + 1$:

$if\ 6 \le a_0 \le 15$: $\quad i_{left} = 2^n + 1 - i$ ;

*else:* $\quad i_{left} = i$ ;

$j_{left} = 1$;

$\mathrm{a}_{0\_\mathrm{left}} = \mathrm{east}(\mathrm{a}_0)$ ;

*else*:

$i_{left} = i$ ;

$j_{left} = j + 1$;

$\mathrm{a}_{0\_\mathrm{left}} = \mathrm{a}_0$ ;

(3) To find the top or bottom neighbour,

$if\ i = 1$:

$if \left\lfloor \frac{a_0 - 1}{5} \right\rfloor mod\ 2 = 0\ and\ (j\ mod\ 2)! = 0$: $\quad \mathrm{a}_{0\_\mathrm{top}} = \mathrm{north}(\mathrm{a}_0)$ ;

$else\ if \left\lfloor \frac{a_0 - 1}{5} \right\rfloor mod\ 2! = 0\ and\ (j\ mod\ 2)! = 0$: $\quad \mathrm{a}_{0\_\mathrm{top}} = \mathrm{south}(\mathrm{a}_0)$ ;

*else*:

$\mathrm{a}_{0\_\mathrm{top}} = \mathrm{a}_0$ ;

$i_{top} = i + 1$ ;

$j_{top} = j - 1$ ;

$i_{top} = i$ ;

$j_{top} = j$ ;

*else*:

$$if\ (j\ mod\ 2) = 0:$$

$$i_{top} = i + 1\ ;$$

$$j_{top} = j - 1\ ;$$

$$else:$$

$$i_{top} = i - 1\ ;$$

$$j_{top} = j + 1\ ;$$

$$\mathrm{a}_{0\_top} = \mathrm{a}_0\ ;$$

The algorithm for finding neighbour triangle address coordinates is very easy to implement and we only use the level 0 neighbours (Equations (34) to (37)) when dealing with edge or corner triangles. The quaternary and QTS codes are then calculated as before.

For example, the three direct neighbour triangles of Big Ben at level 19 shown in Figure 5 (Blue areas) have QTS codes:

Left neighbour:

$QTS: F49PWP23C - 19$ or $Foxtrot, 4{,}9, Papa, Whiskey, Papa, 2{,}3, Charlie - 19$

Right neighbour:

$QTS: F49PWP23D - 19\ or\ Foxtrot, 4{,}9, Papa, Whiskey, Papa, 2{,}3, Delta - 19$

Top neighbour:

$QTS: F49PWP23B - 19\ or\ Foxtrot, 4{,}9, Papa, Whiskey, Papa, 2{,}3, Bravo - 19$

Finally, the area of a quadrangle defined by two opposite corners ($\phi_1$,$\lambda_1$) and ($\phi_2$,$\lambda_2$) is given by:

$$\int_{\phi_1}^{\phi_2} R \int_{\lambda_1}^{\lambda_2} R\ cos(\phi) d\lambda\, d\phi = \int_{\phi_1}^{\phi_2} R^2[\lambda_2 - \lambda_1] cos(\phi) d\phi = R^2(\lambda_2 - \lambda_1)\big(sin(\phi_2) - sin(\phi_1)\big) \tag{38}$$

## Appendix D

## Discrete Global Grid Systems Bibliography

Amiri AM, Samavati F, Peterson P. 2015. Categorization and conversions for indexing methods of discrete global grid systems. ISPRS International Journal of Geo-Information 4:320-336. doi:10.3390/ijgi4010320

Bailey HP. 1956. Two grid systems that divide the entire surface of the earth into quadrilaterals of equal area. Transactions, American Geophysical Union 37(5):628-635.

Bauer-Marschallinger B, Sabel D, Wagner W. 2014. Optimisation of global grids for high-resolution remote sensing data. Computers & Geosciences 72:84–93.

Baumgardner JR, Frederickson PO. 1985. Icosahedral discretization of the two-sphere. SIAM Journal on Numerical Analysis 22(6):1107-1115.

Ben J, Li YL, Wang R. 2016. Radix representation of triangular discrete grid system. Proceedings of the 6th Digital Earth Summit. 6 pp. Doi:10.1088/1755-1315/46/1/012047

Brooks DR. 1981. Grid systems for earth radiation budget experiment applications. NASA Technical Memorandum 83233. National Aeronautics and Space Administration, Hampton, VA. 42 pp.

Carr D, Kahn R, Sahr K, Olsen T. 1998. ISEA discrete global grids. Statistical Computing & Statistical Graphics Newsletter 8(2/3):31-39.

Dutton G. 1989. Planetary modelling via hierarchical tessellation. Proceedings of the Eleventh International Conference on Computer-Assisted Cartography (Auto-Carto 9). Anderson E, ed. American Congress on Surveying and Mapping, Baltimore, MD. pp 462-471.

Dutton G. 1990. Locational properties of quaternary triangular meshes. Proceedings of the 4th International Symposium on Spatial Data Handling. Department of Geography, University of Zurich. pp 901-910.

Dutton G. 1999. A hierarchical coordinate system for geoprocessing and cartography. Lecture Notes in Earth Science 79. Berlin: Springer-Verlag. XIX + 231 pp. 97 figs., 12 plates, 16 tabs. ISSN 0930-0317; ISBN 3-540-64980-8.

Fekete G. 1990. Rendering and managing spherical data with sphere quadtrees. Proceedings of Visualization '90. IEEE Computer Society, Los Alamitos, CA. pp 176-186.

González Á. 2010. Measurement of areas on a sphere using Fibonacci and latitude-longitude lattices. Mathematical Geosciences 42:49-64.

Goodchild MF, Shiren Y, Dutton G. 1991. Spatial data representation and basic operations on triangular hierarchical data structure. Technical Paper 91-8. National Center for Geographical Information and Analysis, University of California, Santa Barbara, CA. 39 pp.

Goodchild MF, Shiren Y. 1992. A hierarchical spatial data structure for global geographic information systems. Computer Vision, Graphics, and Image Processing 54:31-44.

Gregory MJ, Kimerling AJ, White D, Sahr K. 2008. A comparison of intercell metrics on discrete global grid systems. Computers, Environment and Urban Systems 32(3):188-203.

Heikes R, Randall DA. 1995. Numerical integration of the shallow-water equations on a twisted icosahedral grid. Part I: basic design and results of tests. Monthly Weather Review 123(6):1862-1880.

Heikes R, Randall DA. 1995. Numerical integration of the shallow-water equations on a twisted icosahedral grid. Part II: a detailed description of the grid and an analysis of numerical accuracy. Monthly Weather Review 123(6):1881-1887.

Hou M-l. 2004. The basic topology model of spherical surface digital space. Proceedings 20th ISPRS. 6 pp.

Kiester AR, Sahr K. 2008. Planar and spherical hierarchical, multi-resolution cellular automata. Computers, Environment and Urban Systems 32(3):204-213.

Kimerling AJ, Sahr K, White D, Song L. 1999. Comparing geometric properties of discrete global grids. Cartography and Geographic Information Systems 26(4):271-287.

Kimerling AJ, Sahr K, White D. 1997. Global scale data model comparison. Proceedings of the Thirteenth International Conference on Computer-Assisted Cartography (Auto-Carto 13). American Congress on Surveying and Mapping, Seattle, WA. pp 357-366.

Lee M, Samet H. 1998. Traversing the triangle elements of an icosahedral spherical representation in constant time. Proceedings of the 8th International Symposium on Spatial Data Handling. Poiker TK, Chrisman N, eds. International Geographical Union, Burnaby, BC. pp 22-33.

Lugo JA, Clarke KC. 1995. Implementation of triangulated quadtree sequencing for a global relief data structure. Proceedings of the Twelfth International Conference on Computer-Assisted Cartography (Auto-Carto 12). American Congress on Surveying and Mapping, Charlotte, NC. pp 147-156.

Lukatela H. 1987. Hipparchus geopositioning model: an overview. Proceedings of the Eighth International Conference on Computer-Assisted Cartography (Auto-Carto 8). American Congress on Surveying and Mapping. pp 87-96.

Mark DM, Lauzon JP. 1985. Approaches for quadtree-based geographic information systems at continental or global scales. Proceedings of the Seventh International Conference on Computer-Assisted Cartography (Auto-Carto 7). American Congress on Surveying and Mapping. pp 355-364.

Olsen AR, Stevens DL, Jr., White D. 1998. Application of global grids in environmental sampling. Proceedings of The 30th Symposium on the Interface, Computing Science and Statistics: Computing Science and Statistics, Vol. 30. Minneapolis, MN. Weisberg S, ed. Interface Foundation of North America, Fairfax Station, VA. pp 279-284.

Otoo EJ, Zhu H. 1993. Indexing on spherical surfaces using semi-quadcodes. Proceedings of Third International Symposium on Advances in Spatial Databases. Abel DJ, Ooi BC, eds. Lecture Notes in Computer Science 692. Springer-Verlag, Singapore. pp. 510-529.

Ottoson P, Hauska H. 2002. Ellipsoidal quadtrees for indexing of global geographical data. International Journal of Geographical Information Science 16(3):213-226

Raskin R. 1994. Spatial analysis on the sphere: a review. Technical Paper 94-7. National Center for Geographic Information and Analysis, Santa Barbara, CA. 47 pp.

Sadourny R, Arakawa A, Mintz Y. 1968. Integration of the nondivergent barotropic vorticity equation with an icosahedral-hexagonal grid for the sphere. Monthly Weather Review 96(6):351-356.

Saff EB, Kuijlaars A. 1997. Distributing many points on a sphere. Mathematical Intelligencer 19(1):5-11.

Sahr K. 2008. Location coding on icosahedral aperture 3 hexagon discrete global grids. Computers, Environment and Urban Systems 32(3):174-187.

Sahr K. 2011. Hexagonal discrete global grid systems for geospatial computing. Archives of Photogrammetry, Cartography and Remote Sensing 22:363-376.

Sahr K, White D. 1998. Discrete global grid systems. Proceedings of the 30th Symposium on the Interface, Computing Science and Statistics 30:269-278.

Sahr K, White D, Kimerling AJ. 2003. Geodesic discrete global grid systems. Cartography and Geographic Information Science 30(2):121-134.

Satoh M, Matsuno T , Tomita H, Miura H, Nasuno T, Ig S. 2008. Nonhydrostatic icosahedral atmospheric model (NICAM) for global cloud resolving simulations. Journal of Computational Physics 227:3486–3514.

Schröder P, Sweldens W. 1995. Spherical wavelets: efficiently representing functions on the sphere. Proceedings, SIGGRAPH95. Association for Computing Machinery. pp 161-172.

Song L, Kimerling AJ, Sahr K. 2002. Developing an equal area global grid by small circle subdivision. Discrete global grids: a web book. Goodchild MF, Kimerling AJ, eds. University of California, Santa Barbara. URL: https://escholarship.org/uc/item/9492q6sm.

Stuhne GR, Peltier WR. 1999. New icosahedral grid-point discretizations of the shallow water equations on the sphere. Journal of Computational Physics 148:23–58.

Swinbank R, Purser RJ. 2006. Fibonacci grids: a novel approach to global modeling. Quarterly Journal of the royal Meteorological Society 132:1769-1793.

Tegmark M. 1996. An icosahedron-based method for pixelizing the celestial sphere. The Astrophysical Journal 470:L81–L84.

Thuburn J. 1997. A PV-based shallow-water model on a hexagonal-icosahedral grid. Monthly Weather Review 125:2328-2347.

Tobler W, Chen Z-T. 1986. A quadtree for global information storage. Geographical Analysis 18(4):360-371.

Tomita H, Satoh M, Goto K. 2002. An Optimization of the icosahedral grid modified by spring dynamics. Journal of Computational Physics 183:307–331.

White D. 2000. Global grids from recursive diamond subdivisions of the surface of an octahedron or icosahedron. Environmental Monitoring and Assessment 64(1):93-103..

White D, Kimerling AJ, Overton WS. 1992. Cartographic and geometric components of a global sampling design for environmental monitoring. Cartography and Geographic Information Systems 19(1):5-22.

White D, Kimerling AJ, Sahr K, Song L. 1998. Comparing area and shape distortion on polyhedral-based recursive partitions of the sphere. International Journal of Geographical Information Science 12(8):805-827.

Wickman FE, Elvers E, Edvarson K. 1974. A system of domains for global sampling problems. Geografiska Annaler, Series A 56(3-4):201-212.

Williamson DL. 1968. Integration of the barotropic vorticity equation on a spherical geodesic grid. Tellus, 20(4), 642–653.

Yan J, Song X, Gong G. 2016. Averaged ratio between complementary profiles for evaluating shape distortions of map projections and spherical hierarchical tessellations. Computers & Geosciences 87:41–55.

Zhou YM. 1998. Equidistribution and extremal energy of N points on the sphere. Modelling and computation for applications in mathematics, science, and engineering. Jerome JW, ed. Oxford University Press, New York.