Chapter 10
Map Projections

10.1 Introduction to Datums, Coordinate Systems, and Projections

Any system that uses maps must take into account three properties: datums, coordinate systems, and projections. These properties specify our model of the Earth and the way in which we specify locations upon it. In the world of geodesy there are a number of different options for each of these properties with no one "best fit" choice for all applications. While it is often not necessary to use more than one datum or coordinate system, it is important to understand them in case there is a need for interoperability with another system or data source which uses a different datum or coordinate system. As such, we will provide a general introduction to these two properties.

This chapter will focus primarily on map projections, the means by which the three-dimensional Earth is represented in two-dimensional map images. While we will provide a basic introduction to map projections, the focus of this chapter will be on converting between different map projections and how projections are used in a tile-based mapping system.

10.1.1 The Shape of the Earth

Before we can discuss referencing locations on the surface of the Earth or projecting the surface onto a flat plane, we must define the shape of the Earth. The technical term for the true shape of the Earth is the geoid. Formally, the geoid is a surface where gravity's strength is equal at mean sea level. For our purposes it is only important to note that a geoid is not a flat surface but has variations over the surface of the

J.T. Sample and E. Ioup, *Tile-Based Geospatial Information Systems: Principles and Practices*, DOI 10.1007/978-1-4419-7631-4_10,

Earth. Figure 10.1[1] shows gravity variations over the Earth's surface. Figure 10.2[2] shows the variations between the geoid and the most common approximation of the shape of the Earth. The earliest approximations of the geoid used a sphere as the shape of the Earth. In fact, a sphere is often used today to approximate the shape of the Earth. The errors of a spherical approximation are not visible at whole Earth scales.

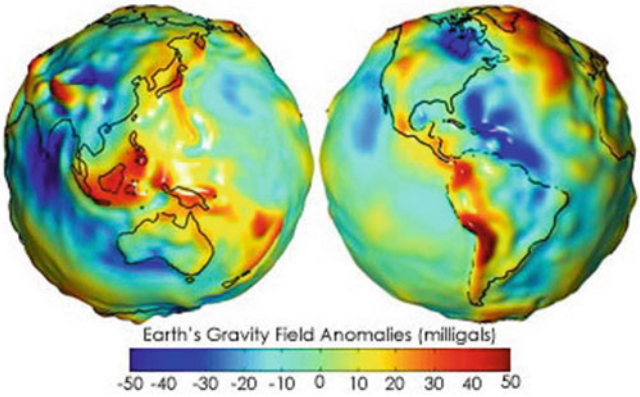


Fig. 10.1 Gravity variations over the surface of the Earth.

A sphere is not the best approximation of the Earth because of flattening at the poles. Instead, the best approximation is the elliptical analog to a sphere, called either an ellipsoid or spheroid (see Figure 10.3). A number of different ellipsoid approximations are currently in use. Different ellipsoids may provide better approximations for specific areas of the Earth. There are also a number of ellispoids which have been superceded but still can be found in older maps or measurements.

10.1.2 Datums

Individual ellipsoids are often referred to in the context of a datum. A datum is an ellipsoid along with an origin point from which locations are referenced (see Figure 10.4). Datums may be global and cover the entire Earth, or they may be local and designed for mapping over a very small area. We are most interested in global datums for the purpose of tile-based mapping; however, it is possible that a local datum will be encountered in source data. Converting between datums is possible,

[1] Courtesy of NASA. http://earthobservatory.nasa.gov/Features/GRACE/page3.php

[2] http://commons.wikimedia.org/wiki/File:Geoid_height_red_blue.png

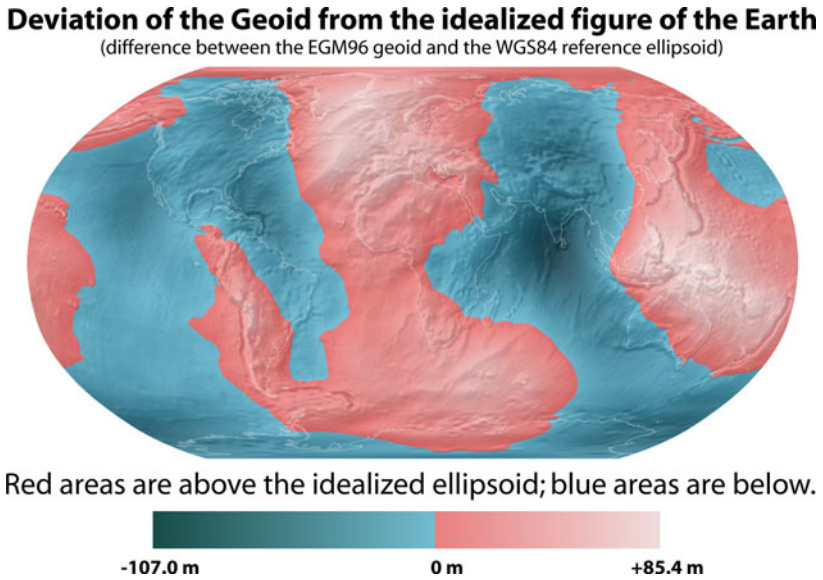


Fig. 10.2 Deviation of the geoid from the shape of the most commonly used ellipsoid approximation.

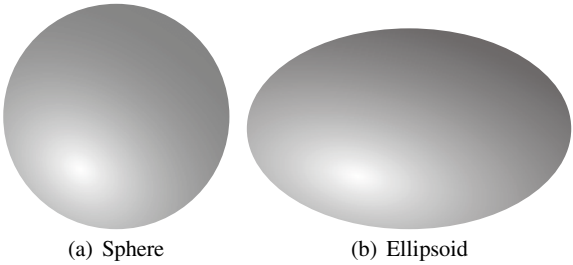


Fig. 10.3 The sphere and ellipsoid are commonly used as approximations for the shape of the Earth. The above ellipsoid has an exaggerated flattening to highlight its shape. Earth approximating ellipsoids visually resemble a sphere because they have only a small amount of flattening.

and the references at the end of the chapter are a good starting point if information on how to perform the conversions is necessary.

The most commonly used datum is the World Geodetic System 1984, commonly abbreviated WGS84. The WGS84 datum has a corresponding ellipsoid that provides a good approximation for the entire Earth. While locally oriented ellipsoids will provide better approximations for small areas, the WGS84 ellipsoid was designed to be a "best fit" for the entire Earth. The origin for the WGS84 datum is the center of the Earth, rather than a point on the surface as is common for many local datums. This geocentric origin is necessary for satellite systems to use the datum.

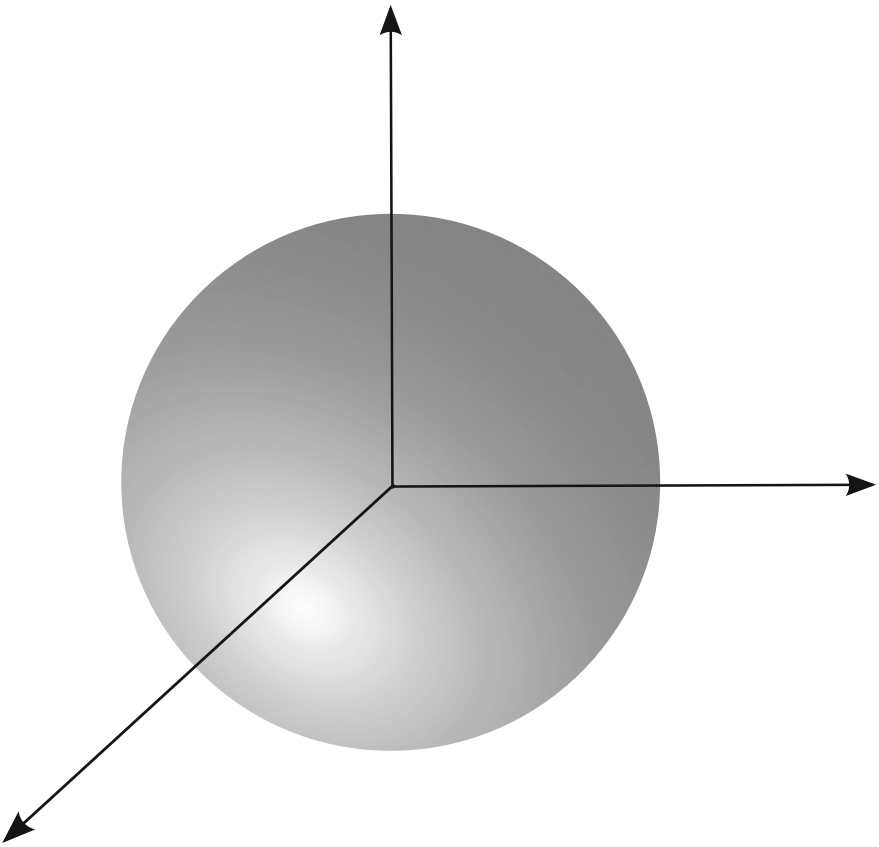


Fig. 10.4 A datum is both an approximation for the Earth's shape as well as an origin.

As stated above, WGS84 is by far the most commonly used datum for geospatial data. Most source data used in a tile-based mapping system will use the WGS84 datum. Some exceptions are the NAD27 or WGS72 datums, both obsolete datums once in use in the United States. The only reason to use a local datum is when a mapping application requires greater local geoid accuracy than the WGS84 ellipsoid can provide. This level of accuracy is rarely needed in a tile-based mapping system. On the other hand, the WGS84 datum provides the most compatibility with other software applications and data sources. Interoperability is an increasingly important component of mapping systems. Thus, in most cases, a tile-based mapping system should always use the WGS84 datum for native backend-data. The added benefit of using WGS84 is that it is so common that most data will already be in this datum with no additional work [2].

10.1.3 Coordinate Systems

Once a datum is selected to represent the shape the Earth, a coordinate system must be defined to specify locations on its surface. There are two coordinate system types in use today. The first is the commonly used geographic coordinate system which uses latitude and longitude to specify a location. Geographic coordinates are a basic angular coordinate system, either over a sphere or ellipsoid depending on which approximation is used. The latitude is the angle north or south from the equator in the range -90 to 90 degrees. The longitude is the angle east or west from the Central (Greenwich) meridian in the range -180 to 180 degrees.

An alternate way of specifying locations on the surface of the Earth is with rectangular coordinates. Rectangular coordinates are a Cartesian coordinate system with X and Y coordinates to represent horizontal and vertical position, respectively. Because a Cartesian coordinate system is used for a two-dimensional grid, rectangular coordinates are only used on flat maps. Usually, the coordinates are specified using meters. As a result, they are useful for performing calculations, such as distance measurements, where the angular geographic coordinates would be cumbersome. Rectangular coordinates are rarely used for maps of small scale, i.e. covering a wide area of the Earth. The distortions associated with the flattening of the Earth become so large that calculations with rectangular coordinates would be of no use. Thus, rectangular coordinates are limited to maps of smaller areas such as topographic maps or high-resolution aerial imagery [6].

10.2 Map Projections

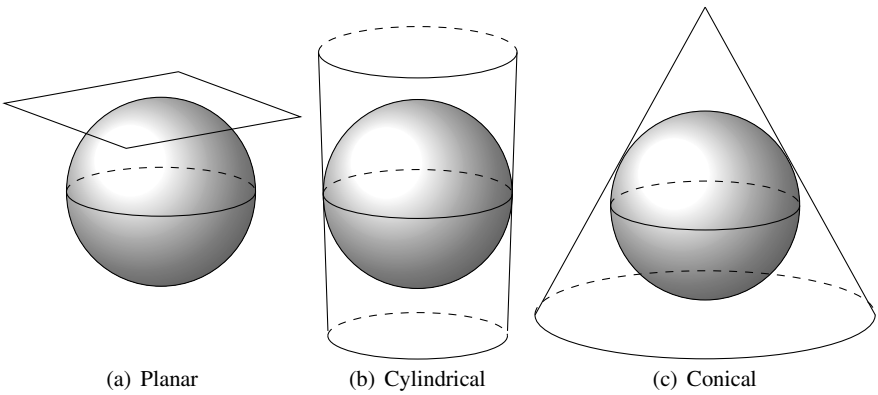
As discussed earlier, the Earth is a geoid which is either approximated as a sphere or ellipsoid. Using two-dimensional maps of the Earth, such as a paper map or satellite image, requires transforming the surface of the Earth to a plane, called a projection. Unfortunately, transforming the surface of a sphere onto a plane causes distortion. It is mathematically impossible to design a projection which does not cause some type of distortion. As a result, there are a number of different types of projections in common usage which limit one type of distortion in exchange for increasing others. The discussion of specific map projections in this chapter will be limited to common projections most likely to be encountered by the implementor of a tile-based mapping system.

Whenever the surface of the Earth is projected onto a plane there is distortion. A number of different types of distortion can occur depending on which map projection is used, including distortions of area, shape, distance, direction, and angle. Individual map projections often reduce or eliminate one of these distortion types. For example, Albers' equal-area projection removes distortion of area on the map at the cost of increasing distortion in other areas [7].

For tile-based mapping there are three important types of maps which are important to recognize. The first type is the equidistant map which most commonly

preserves the length of the meridians. Each line of longitude is of the same length in an equidistant map, so that distances measured on a line of longitude are the same as on the globe. The second type of map is equal-area which preserves the area occupied by a feature. The third map type is conformal which preserves the shape of features. An additional property of conformal projections is that any straight line on the map forms an angle of constant bearing with each line of longitude. This latter property is useful for navigation and led to the historic popularity of conformal projections, such as the Mercator projection.

A projection may be created by directly mapping the globe to a planar surface. However, it is also possible to perform the trasformation by first projecting the globe to an intermediate shape. The two most common intermediate shapes are a cylinder and a cone. Both the cylinder and the cone may be transformed into a plane with no distortion. While not all projections are formed in this manner, many of the most commonly used projections are either cylindrical, conic, or planar, as shown in Figure 10.5. Distortion is always minimal at the location where the shape touches the globe, such as the equator for many cylindrical projections [6].



(a) Planar (b) Cylindrical (c) Conical

Fig. 10.5 Projections are created by projecting Earth's surface onto a surface that may be transformed into a plane.

10.2.1 Different Map Projections

Theoretically, there are an infinite number of different map projections. In practice, there are merely dozens in common usage on paper maps. For the purpose of tile-based mapping, there are only a few map projections which are important to discuss. Tile-based mapping systems, or any computer mapping system, must focus heavily on interoperability. Thus, the projections used in these systems tend to be limited to

a few standard projections. In contrast, paper maps do not need to interoperate with any other data or system and may select the projection with the best utility for the map's specific view of the Earth.

10.2.2 Cylindrical Equidistant Projection

The first important projection for tile-based mapping is the cylindrical equidistant projection. This projection is sometimes called the Cylindrical Equirectangular, Plate Carrée, Simple Cylindrical, WGS84 Geodetic, or WGS84 Lat/Lon projection [7]. The cylindrical equidistant projection does not scale the meridians of the original globe. Thus, distances are not distorted north-to-south. However, each line of latitude is stretched to the same length as the equator, providing significant stretching east-to-west. The distortion in this projection ranges from 0 at the equator to infinity at the poles. Additionally, neither area, shape, nor bearing is preserved on the map. Figure 10.6 is an example of a geodetic map of the world.

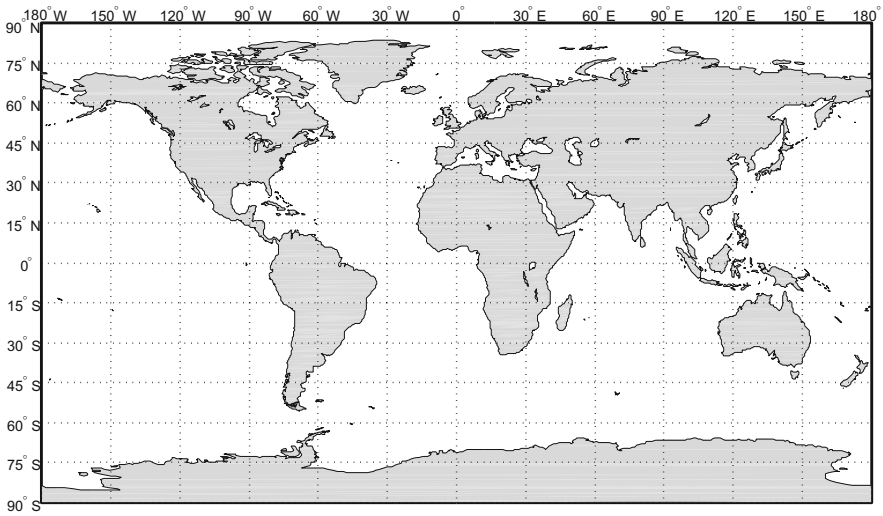


Fig. 10.6 Geodetic projected map of the world.

The benefit of this map projection is the ease of construction, especially for computer mapping systems. One degree of latitude and one degree of longitude are the same length in any area of the map. The same is not true of other projections (see Mercator projection below). As a result, the resulting map forms a simple Cartesian coordinate system of the world, centered at longitude and latitude (0,0). Another useful consequence is that each pixel in a map image represents the same distance

in both the east-west and north-south directions, simplifying calculations performed on map images.

10.2.3 Cylindrical Equal-Area Projection

The cylindrical equidistant projection is simple and easy to use; however, it offers little beneficial map properties other than simplicity. The parallels are stretched by a value of $\sec\lambda$, where λ is the latitude. By scaling the projection by a factor of $\cos\lambda$, we can create the cylindrical equal-area projection. The cylindrical equal-area projection preserves the area of features but not the distances between them or angles on the map.

The cylindrical equal-area projection is a rarely used projection, in both printed maps and computer map images. However, it is useful when used with data in the cylindrical equidistant projection. Maps at large scale can be reprojected to equal-area by multiplying by the cosine of the average latitude of the map. Reprojecting to equal-area projection at display time can reduce visible distortions of features.

10.2.4 Mercator

The Mercator projection is a cylindrical conformal projection where the cylinder touches the globe at the equator. A conformal projection preserves the angles and shapes of features on the map. This property makes conformal projections popular in traditional mapping and cartography. Historically, conformal maps were useful for navigation because straight lines on the map have constant bearing. Ships would not have to change bearing in order to follow a straight route on a conformal map. Land surveyors find a conformal projection useful because angles measured on the ground can be transfered directly to the map for use in computation.

The Mercator projection is one of the oldest known map projections, dating from the 16^{th} century [7]. Similar to the previous two cylindrical map projections, the Mercator projection has equally spaced and equal length lines of longitude. Lines of latitude are also of equal length but are not equally spaced. Distance between lines of latitude increases away from the equator. The result is a distinct increase in size of features towards the poles. The features themselves are the same shape as on the globe. Figure 10.7 is an example of a Mercator map of the world.

10.2.5 Universal Transverse Mercator

The Transverse Mercator projection is a cylindrical projection similar to the standard Mercator projection. However, rather than deriving the projection from a cylin-

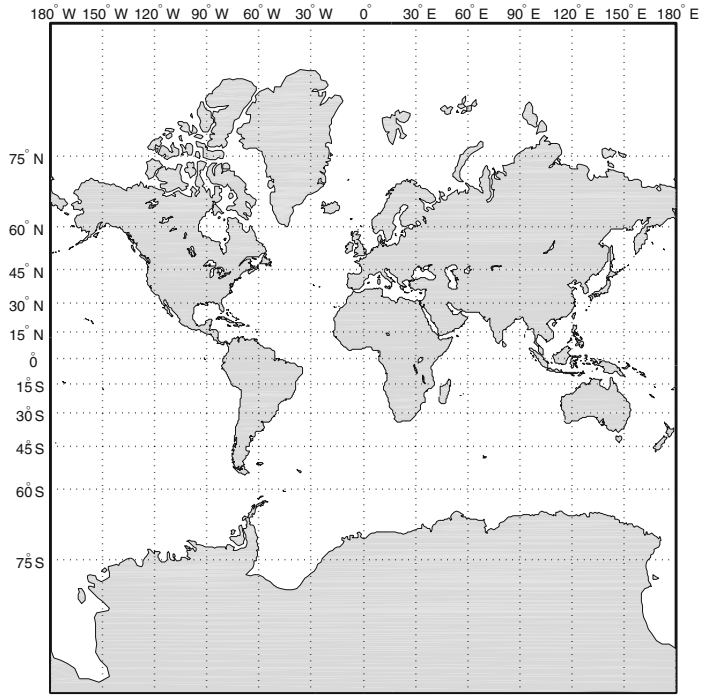
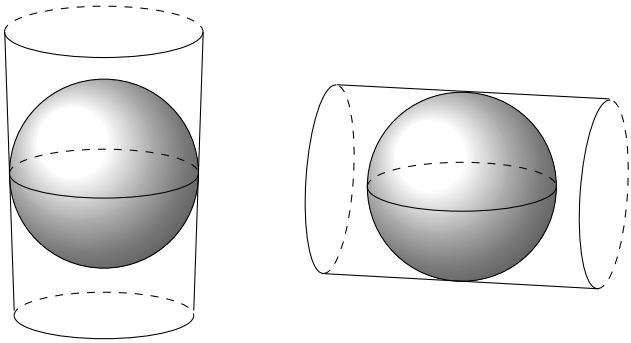


Fig. 10.7 Mercator projected map of the world.

der oriented vertically pole to pole, the Transverse Mercator projection is based on a cylinder oriented horizontally parallel to the equator, as shown in Figure 10.8. Lines of latitude and longitude are no longer straight in the Transverse Mercator projection. Whereas the Mercator projection distorts features farther north or south, the Transverse Mercator projection distorts features lying farther east or west from the central meridian. The projection can be made using any line of longitude as the central meridian and provides little distortion to areas within a short range. As a result, the Transverse Mercator projection is often used for maps of small scale for use in land surveying. Transverse Mercator maps primarily use rectangular coordinates (meters) to simplify ground calculations over small areas.

Universal Transverse Mercator (UTM) is a standard coordinate and projection system commonly used for imagery, topographic maps, and other low scale geospatial data. The UTM system covers the entire world from 80°S to 84°N. Each hemisphere is split into 60 zones 6° wide. Zones are number 1 to 60 from west to east starting at 180°W. Zones are identified by their zone number and hemisphere (North or South).

Each UTM zone has its own specific projection and coordinate system. The central longitude of each zone is used as the central meridian for a Transverse Mercator projection. UTM uses a rectangular coordinate system with the origin at the intersection of the central meridian and the equator. Locations are identified by their



(a) Mercator (b) Transverse Mercator

Fig. 10.8 Mercator and Transverse Mercator both project the Earth's surface onto a cylinder. The difference between the two is the orientation of the cylinder.

easting and northing values , the distance east and north in meters from the zone origin. However, in order to prevent negative easting and northing values a constant is added to all easting and northing values. This constant is called a false easting or false northing. UTM specifies a false easting of 500,000m in both hemispheres and a false northing of 0m in the Northern Hemisphere and 10,000,000m in the Southern Hemisphere. For example, in Zone 1 South, the intersection of the central meridian (174°W) and the equator is given the coordinate 500000E, 10000000N. Figure 10.9 shows the coordinates of two points before and after the conversion to false northing and easting.

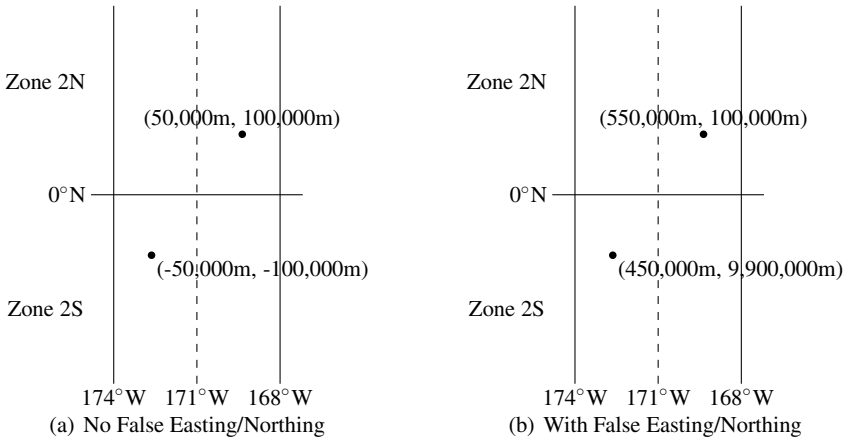


Fig. 10.9 False northing and easting values are simply an artificial translation of the coordinates to ensure they are non-negative.

Since UTM zones are only 6° wide, the amount of spatial distortion on any UTM map is small. As a result, UTM is used frequently in satellite and aerial imagery. Source data used in tiled map systems will often be projected using UTM. For example, USGS produced Digital Orthophoto Quadrangle (DOQQ) aerial imagery is distributed using UTM.

10.3 Point Reprojection

Often, it is necessary to convert between projections in the process of creating a tiled-mapping system. Source data will often be distributed using a projection such as UTM while data in the tiled-mapping system will be stored natively in a global projection such as Lat/Lon or Mercator. We will present some basic reprojection formulae and techniques and demonstrate the process of converting from UTM to Lat/Lon.

Before covering techniques for reprojecting images, we must explain how to reproject individual points. We will present the forumlas here along with some example code. The formulae and a more thorough explanation can be found in Snyder's "Map Projections: A Working Manual" [7].

Below are the formulae for converting a Geodetic coordinate into a UTM coordinate:

$$x = X_0 + k_0 N \left[A + (1 - T + C) \frac{A^3}{6} + \left(5 - 18T + T^2 + 72C - 58e'^2\right) \frac{A^5}{120} \right] \tag{10.1}$$

$$y = Y_0 + k_0 \left\{ M - + N \tan\phi \left[\frac{A^2}{2} + \left(5 - T + 9C + 4C^2\right) \frac{A^4}{24} + \left(61 - 58T + T^2 + 600C - 330e'^2\right) \frac{A^6}{720} \right] \right\} \tag{10.2}$$

where

$$e = \sqrt{1 - \frac{b^2}{a^2}} \tag{10.3}$$

$$e'^2 = \frac{e^2}{(1 - e^2)} \tag{10.4}$$

$$N = \sqrt{\frac{a}{1 - e^2 \sin^2\phi}} \tag{10.5}$$

$$T = \tan^2\phi \tag{10.6}$$

$$C = e'^2 \cos^2\phi \tag{10.7}$$

$$A = (\lambda - \lambda_0)\cos\phi \tag{10.8}$$

$$\begin{aligned} M = a[&(1 - e^2/4 - 3e^4/64 - 5e^6/256 - \ldots)\phi - (3e^2/8 + 3e^4/32 + 45e^6/1024 + \ldots)\sin^2\phi \\ &+ (15e^4/256 + 45e^6/1024 + \ldots)\sin^4\phi - (35e^6/3072 + \ldots)\sin^6\phi + \ldots] \end{aligned} \tag{10.9}$$

and

ϕ = latitude (radians)
λ = longitude (radians)
λ_0 = central longitude of the UTM zone (radians)
$a = 6,378,137$
$b = 6,356,752.3$
$k_0 = 0.9996$
$X_0 = 500,000$

$$Y_0 = \begin{cases} 0 & \phi \geq 0 \\ 10,000,000 & \phi < 0 \end{cases}$$

Code for transforming a point from Geodetic to UTM is given in Listing 10.2.

The inverse formulae for converting from UTM to Geodetic coordinates are below. Variable definitions are equivalent to the forward conversion formulae unless otherwise noted.

$$\phi = \phi_1 - \left(N_1 \frac{\tan\phi_1}{R_1}\right)\left[\frac{D^2}{2} - \left(5 + 3T_1 + 10C_1 - 4C^2 - 9e'^2\right)\frac{D^4}{24} + \left(61 + 90T_1 + 298C_1 + 45T_1^2 - 252e'^2 - 3C_1^2\right)\frac{D^6}{720}\right] \tag{10.10}$$

$$\lambda = \lambda_0 + \left[D - (1 + 2T_1 + C_1)\frac{D^3}{6} + \left(5 - 2C_1 + 28T_1 - 3C_1^2 + 8e'^2 + 24T_1^2\right)\frac{D^5}{120}\right]\frac{1}{\cos\phi_1} \tag{10.11}$$

$$\phi_1 = \mu + \left(\frac{3e_1}{2} - \frac{27e_1^3}{32} + \ldots\right)\sin 2\mu + \left(\frac{21e_1^2}{16} - \frac{55e_1^4}{32} + \ldots\right)\sin 4\mu \tag{10.12}$$

$$+ \left(\frac{151e_1^3}{96} - \ldots\right)\sin 6\mu + \left(\frac{1097e_1^4}{512} - \ldots\right)\sin 8\mu + \ldots \tag{10.13}$$

where

$$e_1 = \frac{1 - \sqrt{1 - e^2}}{1 + \sqrt{1 - e^2}} \tag{10.14}$$

$$\mu = \frac{M}{a\left(1 - \frac{e^2}{4} - \frac{3e^4}{64} - \frac{5e^6}{256} - \ldots\right)} \tag{10.15}$$

$$M = \frac{y - Y_0}{k_0} \tag{10.16}$$

$$e'^2 = \frac{e^2}{(1 - e^2)} \tag{10.17}$$

$$C_1 = e'^2 \cos^2\phi_1 \tag{10.18}$$

$$T_1 = \tan^2\phi_1 \tag{10.19}$$

$$N_1 = \sqrt{\frac{a}{1 - e^2\sin^2\phi_1}} \tag{10.20}$$

$$R_1 = \frac{a(1 - e^2)}{(1 - e^2\sin e^2\phi_1)^{\frac{3}{2}}} \tag{10.21}$$

$$D = \frac{x - X_0}{N_1 k_0} \tag{10.22}$$

Both the forward and inverse conversion formulae are based on series expansion approximations. The expansions converge quickly and are accurate to about a centimeter given the 6° width of a UTM zone. It is important to note the extensive floating point math in these formulae. Using double-precision floating point math

with these formulae is quite expensive. Performing a single Geodetic to UTM conversion will appear instantaneous on modern day computers. However, when scaled to thousands, or even millions, of conversions, the costs become noticeable.

10.4 Map Reprojection

The previous section dealt only with converting single locations from one projection to another. While important, individual point reprojection is not the end goal for tiled-mapping. What we really must accomplish is reprojection of entire map images. Reprojecting an entire map image is not a simple matter of applying formulae. We must weigh a number of considerations when determining what techniques to use for image reprojection.

Map images are often large and reprojecting them can be computationally costly. Combined with the large numbers of images in many map image datasets, the issue of algorithm efficiency becomes important. On the other hand, we have the problem of geospatial accuracy. First, we must ensure that a feature in the map image has coordinates representing where it actually is on the globe. The UTM/Geodetic point reprojection formulae presented above do have high accuracy given the 6° width of UTM zones . However, there are a number of different method of doing image reprojection, some of which sacrifice accuracy in exchange for improved performance. In many cases, it is acceptable to sacrifice accuracy in a tiled mapping system. Centimeter accuracy is not necessary in an online mapping system for the layperson. There is another type of distortion which can be highly damaging in a tiled-mapping system: border distortion, the discontinuities between images, is unacceptable in a tiled-mapping system. No level of user will accept discontinuities between tiles. As such, it is imperative that our reprojection technique eliminate visible discontinuities at the edges of images.

10.4.1 Affine Transforms

Affine transforms provide one possible means of reprojecting map images [2]. An affine transform is a linear transformation plus a translation.

$$u = a_0 + a_1 x + a_2 y$$
$$v = b_0 + b_1 x + b_2 y$$

The linear transform in an affine transform may be a combination of rotation, scaling, or shear. Affine transforms preserve collinearity of points. Straight lines before the transform are also straight after the transform. Affine transforms may be represented as a matrix and are computationally simple to apply to an entire image because the transform is linear. Many image libraries provide the functionality of

applying an arbitrary affine transform to an image (e.g., Java, Python Imaging Library).

To reproject an image from UTM to Geodetic using an affine transform, we must determine the parameters of the transform and then apply the transform to the image. Determining the parameters requires solving a simple matrix equation:

$$\begin{pmatrix} 0 & w-1 & w-1 \\ 0 & 0 & h-1 \\ 1 & 1 & 1 \end{pmatrix} = \begin{pmatrix} a & b & c \\ d & e & f \\ 1 & 1 & 1 \end{pmatrix} \begin{pmatrix} x_1 & x_2 & x_3 \\ y_1 & y_2 & y_3 \\ 1 & 1 & 1 \end{pmatrix}$$

The variables w and h are the width and height of the reprojected image. The (x_1, y_1), (x_2, y_2), and (x_3, y_3) are the north-west, north-east, and south-east corner pixel coordinates of the Lat/Lon quad inside the UTM image. These are the pixel locations of the corner points of the Lat/Lon image individually reprojected into the UTM image (using the point reprojection formular using Equations 10.1 and 10.2). The affine transform is the solution defined by the variables a, b, c, d, e, and f. When applied to the UTM image, the three image corner points used to solve the equation will map perfectly to the Lat/Lon image. The final corner will be slightly displaced from the corner in the final image, but the difference will be small in a high resolution image.

The computational efficiency of the affine transform, combined with its ease of use in most programming environments, makes it a good candidate as a reprojection algorithm. For individual map images it is a good method of reprojection, especially if processing power is minimal and measurement distortion is not an issue.

The performance of affine transform reprojection makes it appear to be a great technique for tiled-mapping where large datasets demand efficient algorithms. However, approximating the reprojection with a linear transform causes significant border distortion. As seen in the point reprojection formulae (Equations 10.10 and 10.11), the UTM to Geodetic reprojection is non-linear. Visually, this means that the UTM grid lines and image will become curved. However, the affine transform's collinearity property ensures that they remain straight. Thus, the resulting image is not on a Geodetic grid but slightly distorted. Additionally, each reprojected image uses a unique affine transform best approximating the specific reprojection of that image. The different transforms will cause each image to be reprojected differently. These combined error sources result in significant discontinuities between reprojected images. At high resolution discontinuities in features such as roads in a tiled-mapping system are unacceptable for users and must be avoided. Thus, the affine transform does not provide an acceptable reprojection solution [3].

10.4.2 Interpolation

The alternatives to affine transform reprojection of images are techniques that combine point reprojection with interpolation. The interpolation algorithms can be confusing, so we will explain them below.

Interpolation is a technique to approximate data values that lie within some set of known data values. Linear interpolation is a commonly used and simple interpolation method. It is based on the assumption that the underlying function modeling the data values is linear. As a simple example, if we have known data values $(2,10)$ and $(3,20)$, then using linear interpolation, we can approximate values $(2.5,15)$, $(2.7,17)$, $(2.1,11)$, etc.

But how do we interpolate values of image pixels? Here, we have pixel location represented by a two-dimensional coordinate and color value. Rather than interpolate between two pixels lying on a straight line, we interpolate between four pixels forming a square. Bilinear interpolation provides a means of determining the value (e.g., color) of any pixel lying within any four pixels with known values.

$$f(x,y) = f(0,0)(1-x)(1-y) + f(1,0)x(1-y) + f(0,1)(1-x)y + f(1,1)xy$$

The four known pixels are normalized to coordinates $(0,0)$, $(0,1)$, $(1,0)$, and $(1,1)$. The value of a pixel is represented by the function $f(x,y)$. As seen in the formula, the value of the internal pixel is weighted more heavily towards the pixels it is closest to. For RGB color pixels, the bilinear interpolation must be performed three times, one for each color component. A visualization of bilinear interpolation is shown in Figure 10.10.

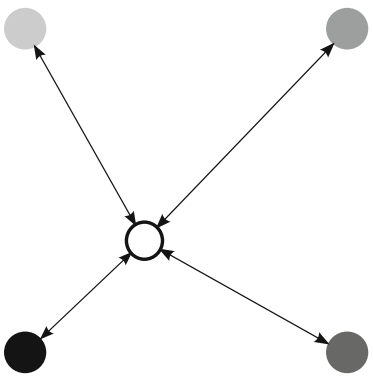


Fig. 10.10 Bilinear interpolation between four points to calculate the value of the interior point.

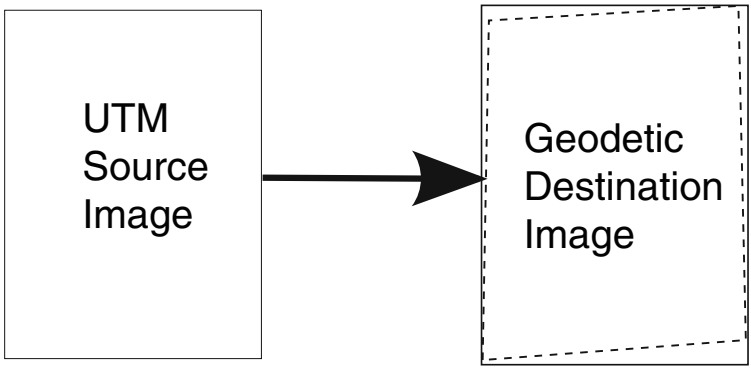
10.4.3 Point-wise Reprojection

The point reprojection formulae presented earlier imply a simple method for reprojecting an entire map image: take every point in the UTM image and reproject it to a Geodetic image. Such an algorithm would work, except that a pixel in the UTM image is highly unlikely to be reprojected exactly onto a pixel in the Geodetic image. Instead, it will map to a point between pixels. One could devise a method of reprojecting UTM points and then performing interpolation to determine values of the Geodetic pixels. However, this would become needlessly complicated and probably lead to a loss of accuracy.

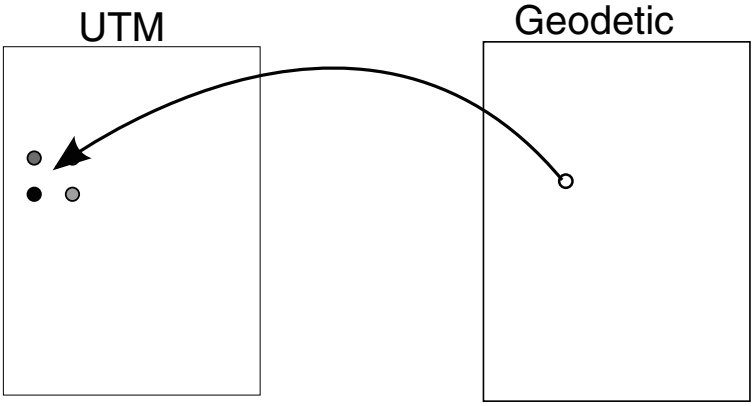
There is a better way of performing point-wise reprojection diagrammed in Figure 10.11. First, start by creating an empty Geodetic image covering the same geographic area as the UTM image. Use the UTM to Geodetic formulae to determine the Geodetic coordinates of the corners of the UTM image. The Geodetic image is the bounding rectangle around those four corners. Then for each pixel in the Geodetic image, reproject the coordinates back into UTM. In general, the UTM coordinates will not correspond to one pixel in the UTM image. Instead, it will lie inside a square bounded by four pixels. The color value for this UTM point can be calculated using bilinear interpolation using the surrounding four pixels' color values. This color value is also the color value for the pixel in the Geodetic image. Fill in the color and proceed to the next pixel.

Point-wise reprojection has the benefit of being accurate, both geospatially and with border alignment. The reprojection accuracy is the same as with the point reprojection formulae with the additional small error caused by the bilinear interpolation for pixel colors. We can assume color interpolation error is minimal because images usually do not have a lot of high frequencies in the color components. The color variation between individual pixels is so small that the underlying function is essentially linear and thus well approximated using bilinear interpolation. Additionally, the distribution mechanism of a tiled-mapping system will often be a lossy compression algorithm such as JPEG, which performs smoothing on the color. Border alignment has no visible error because adjacent images are reprojected using the same transformation, unlike the affine transformation reprojection.

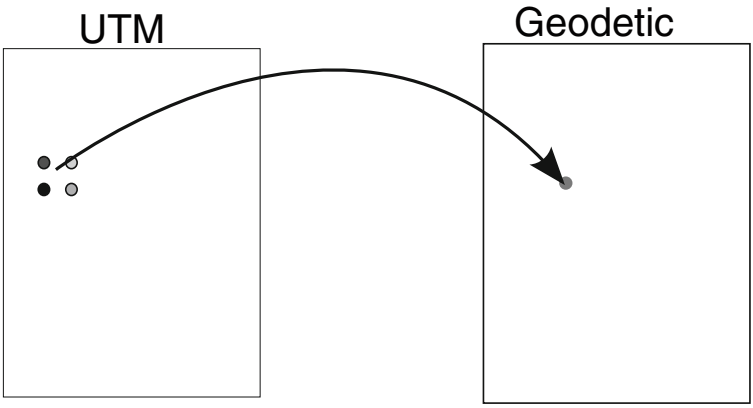
The flip side of the increased accuracy of point-wise reprojection is increased computational cost. The non-linear reprojection formulae are reasonably fast when run once on a modern computer. However, the technique scales poorly. A 10,000 by 10,000 pixel image would require 100,000,000 point reprojections. When that is scaled to the number of images in a global map dataset, the time required to reproject the data becomes untenable. The example code for point reprojection given above takes approximately 20 minutes for 100,000,000 reprojections. Compiled code would be faster but not fast enough. What we really need is an improved algorithm [3].



(a) Calculate the geographic area quadrilateral of the Geodetic image from the UTM source image. Create the Geodetic destination image using the bounding box of the Geodetic area quadrilateral.



(b) Convert the coordinates of a pixel in the Geodetic image to UTM coordinates. The new UTM point will lie between four pixels in the UTM image.



(c) Perform bilinear interpolation using the four UTM pixels to determine the color of the original Geodetic pixel.

Fig. 10.11 Point-wise reprojection of a map. The steps in figure (b) and (c) will be performed for each pixel in the Geodetic image.

10.4.4 Tablular Point-Wise Reprojection

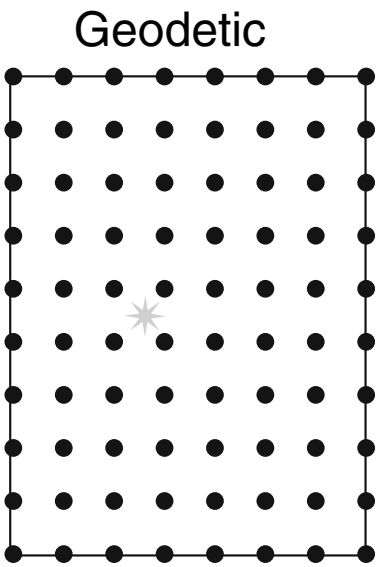
Point-wise reprojection creates excellent alignment of tiled imagery but with a cost of computational complexity. The computation cost is derived from the repeated calculations for each pixel in an image. Reprojecting each pixel provides high geospatial accuracy. However, in general, the accuracy of point reprojection is too high for the images under consideration. The point reprojection algorithms have, at worst, centimeter accuracy, whereas much of the imagery being reprojected has a resolution of 1m per pixel. Thus, centimeter accuracy is unneeded, especially when the computation cost is so high.

Our solution is to reproject only a subset of the points in the image. A table is generated by subsampling the pixels in the Geodetic image. Thus, instead of a 10,000 by 10,000 Geodetic image, we may have a 100 by 100 table covering the same geographic area. Each pixel in the table is projected from Geodetic to UTM. (Remember, in order to convert an image from UTM to Geodetic, the coordinates of each target pixel in the Geodetic image are converted to UTM coordinates so that the target pixel's color may be calculated from the surrounding UTM pixels). The size of the table should be a divisor of the size of the desired Geodetic image and also contain the four corners of the Geodetic image to simplify the algorithms using the table. Once the table is created, it is used in the reprojection of the entire image. To reproject a Geodetic pixel we find the nearest pixels in the table and perform a bilinear interpolation to calculate each UTM component. Reprojecting the entire image requires only linear operations rather than the non-linear reprojection formulae. Figure 10.12 demonstrates the process of table-based reprojection.

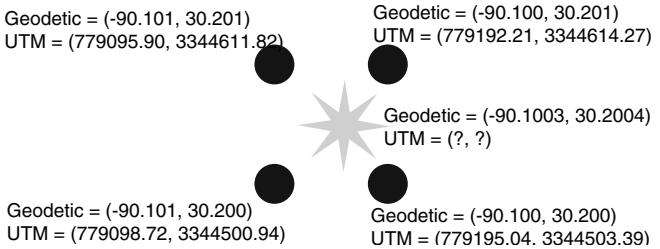
Of course, there is the important question of how tabular reprojection will affect the accuracy of the reprojected image. First, there is no effect on the quality of image tile alignment. The borders of two adjacent images will still be projected using the same method, providing visually perfect alignment. Border alignment is important because discontinuities in a tiled-map system are unacceptable to users.

Geospatial accuracy is also important. The tabular point-wise reprojection will reduce accuracy. The reprojection formulae are non-linear, meaning the linear approximation inherent in the table interpolation will not be exact. The benefit of this method is that the error caused by the linear approximation can be limited by modifying the size of the table. The highest error will occur with a 2x2 table containing true reprojections of only the four corners of the image. A 2x2 table is used by systems to obtain the maximum increase in reprojection speed. Geospatial error will be higher, but border alignment will not be an issue as with any point-wise projection method. The open-source project GDAL (Geospatial Data Abstraction Library) takes this approach. At the low end of the error spectrum is full point-wise reprojection.

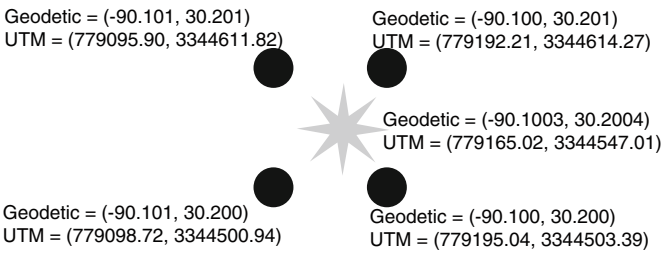
An important fact ignored by either the 2x2 table or full point-wise reprojection methods is the required amount of accuracy needed for the application using the imagery. Imagery with internal 1m accuracy does not require sub 1cm reprojection accuracy. It is a waste of resources to reproject an image with more accuracy than is internal to the image. A better method would be to tailor the table resolution to the



(a) Create the table by performing a UTM reprojection on a subset of the points in the Geodetic image. For other Geodetic points that must be converted, locate their positions in the table.



(b) Perform two bilinear interpolations to calculate the UTM x and y coordinates of the target point.



(c) Use the UTM x and y coordinates in the point-wise image reprojection algorithm.

Fig. 10.12 Table-based reprojection of points in an image.

level of accuracy required for a particular application or dataset. A simple binary search can be used to calculate the necessary resolution. There is no need to create the entire table in the search process, just a representative set of points to test the accuracy. A comparison of values from the true reprojection formulae and the linear interpolation will provide the interpolation error for the given table resolution. In most cases the size of an accuracy-tuned table will not be large. It will be rare that a table larger than 100x100 will be necessary. In our applications, a 16x16 table is adequate to provide our desired geospatial accuracy.

The computational cost of using the table-based approach is much smaller than full point-wise reprojection. Of course, as the size of the table increases, so does the computation cost. However, even a 100x100 table has far fewer non-linear computations in comparison to full point-wise reprojection of a 10000x10000 pixel image. The 2x2 table is highly attractive from a computational cost perspective, but it is important to ensure that the loss of geospatial accuracy is acceptable in the final application. The reduction in computational cost obtained by using a 2x2 table instead of a 16x16 table is limited in comparison to the significant reduction in accuracy [4].

10.5 Map Projections for Tiled Imagery

Once we are able to move between projections, we must decide which Coordinate Reference System (CRS) to use when storing the tiled map imagery. The best datum to use will usually be WGS84. A global tiled-map system will rarely be useful if it is in another datum. WGS84 is the most interoperable datum and the best fit for global datasets.

Choosing a map projection is not so simple. The different projections offer different useful properties. In certain cases, the system will require a particular property, equal area for example, and this will guide the selection of projection. However, we will assume the tiled-map system is primarily used for map browsing and not advanced geospatial applications such as cartography, navigation, etc. In this case, no single projection provides the best solution.

10.5.1 Storing Tiles in the Geodetic Projection

There are two map projections which are most commonly used by tiled-mapping systems . The first is the equidistant cylindrical projection, usually called the Geodetic projection, Lat/Lon projection, or Plate Carrée projection because a latitude and longitude grid is the coordinate system of the projection. In combination with the WGS84 datum, the Geodetic projection is the most used projection for map data provided by Geospatial Web services on the Internet. It is also referred to by its

EPSG code, 4326. The European Petroleum Survey Group has codes for most CRS which has become a standard means of identifying them.

The Geodetic projection is used for Web mapping application so often because it is simple. The coordinate system is latitude and longitude, and the projection forms a perfect grid with those coordinates. Images in geodetic projection are easily overlaid on a globe, such as in 3D map clients such as Google Earth or NASA World Wind. Also, mathematical operations using coordinates, common in map image manipulation, become much simpler with the Geodetic projection. Each pixel in a map image with Geodetic projection has the same dimensions in degrees.

The downside of the Geodetic projection is the distortion. Area, shape, and latitude distances are distorted in the Geodetic projection. These distortions are especially apparent close to the poles. Example distortions are circular domes which become ovals and streets which no longer intersect at right angles. While any map projection creates distortion, these types of distortions are readily apparent to users, especially on a tiled-mapping system designed to display maps at large scale (zoomed in).

As mentioned above, 3D map clients such as Google Earth and NASA World Wind use the Geodetic projection, as well as the online map tool Mapquest. GIS clients understand the Geodetic map projection. WMS clients primarily function using the Geodetic projection.

10.5.2 Storing Tiles in the Mercator Projection

A common alternative to storing tiles in the Geodetic projection is to store them using the Mercator projection . The Mercator is a cylindrical conformal projection, meaning that it preserves angles and shapes on the map. It is also a good projection for global data; most of the globe is clearly visible on a small scale map. While most interoperable geospatial Web services do not use the Mercator projection for their data, it is used in some common tiled-mapping systems on the Web. Google Maps, Yahoo Maps, Microsoft Bing Maps, and Open Street Map all store their tiled imagery in the Mercator projection.

These Web mapping sites use the Mercator projection primarily because of its angle preservation properties. The primary use of these systems is to view map images at a large scale. Users zoom into the map to retrieve street directions or search for locations. The Mercator projection prevents visible distortions in these large scale maps. Road intersections retain the same angles as on the globe. In the Geodetic projection this is not the case. Roads in areas nearer to the poles, such as Finland, intersect at non-right angles. Shapes are also preserved in the Mercator projection. Circular buildings do not become stretched. The primary distortion caused by the Mercator projection, size distortion, is not visible in these large scale maps. At the local level, sizes are uniformly increased in size around the poles, so that the distortions are not visible.

The Mercator projection has downsides for use as a projection for native tile storage. The primary problem for the Mercator projection is the issue of interoperability. Clients which access map services, especially WMS clients, usually support the Geodetic projection. Support for the Mercator projection in non-tile clients is limited. Thus, if the tiled-map system will be a backend to a WMS, using the Mercator projection will limit its use by many clients. Often tiled map data will be used as an overlay or background to other map data. Most of this map data is in the Geodetic projection, and combining with Mercator projected data will be difficult. A good example of this problem is Google Earth. It expects data to be in Geodetic projection. Using Mercator projected data as an overlay in Google Earth will cause problems. Web mapping systems such as Google Maps are not concerned with interoperability. These public Web sites do not support WMS access to their maps, and they discourage the use of their map tiles in other clients. Choosing a projection which increases interoperability at the cost of visible distortion at large map scales is not to their benefit.

10.5.3 Other Projections

There are a number of other possible projections that tiles may be stored in other than Geodetic and Mercator. Each have their beneficial properties. However, any other projection will also reduce the interoperability of the system. While the Mercator projection does reduce interoperability, it does have better tile client support because it is used by the major Web mapping sites. A single tiled-mapping system using a different projection will probably not have as much support from common geospatial data clients. A non-Geodetic projection will also be more difficult to work with, a problem mentioned with the Mercator projection.

If these problems are not an issue for a tiled-mapping system, there are a number of possible projection which can be used. Primary focus should be on global projections. Most conic projections are not designed to view the entire world so they are less useful for a tiled-mapping system. An equal-area projection, such as the Peter's projection, will create a global map without the area distortions of the Mercator projection or Geodetic projection [2]. The Winkel Tripel projection is the global projection used by the National Geographic Society for their world maps since 1998 [1]. It is neither equal-area nor conformal. Instead, its aim is to reduce, but not eliminate, all map distortions: area, angles, and distance. The Robinson projection is similar to the Winkel Tripel and aims to reduce all distortion rather than eliminate one in particular [5].

It should be noted that UTM is not suitable for a tiled-mapping system. While UTM is a global CRS, it is not a global projection. The different projections defined by UTM for the different sections of the globe will cause serious difficulties for a tiled-mapping system. Tiles will most likely cross UTM boundaries causing great projection distortions, especially at small map scale. Even at large map scales, tiles in different projections will be adjacent and cause discontinuities between tiles.

Only tiled-mapping systems with imagery covering an area completely within a UTM zone should consider using UTM for imagery.

It is also possible to use multiple projections for different tile map scales. For small map scales, use a projection such as Winkel Tripel that does not cause significant size distortion, and for large map scale, use a conformal projection such as Mercator which maintains angles. While possible, this would destroy any interoperability because the tiled-map coordinate system would change depending on map scale. For situations where interoperability is not an issue, this method is viable though it would create significant implementation difficulties.

10.5.4 Which Projection for a Tiled-Mapping System?

Given the importance of interoperability in a tiled-mapping system, we recommend the Geodetic projection as the native projection for image tiles. Clients such as Google Earth and World Wind require input data in the Geodetic projection. Most WMS clients expect map data to be in the Geodetic projection. Other geospatial data, which may be used as image overlays, is often in the Geodetic projection. The simplicity of the Geodetic projection provides additional impetus to prefer it over other projections. The latitude/longitude Cartesian coordinate system makes image manipulation simple, which is important for implementing a WMS front end to the system.

While we recommend the Geodetic projection as the native projection for tiles, it is still possible to provide support for Mercator tiles. The transformation between the Geodetic projection and the Mercator projection is much less complex than Geodetic/UTM reprojections. Rather than store the tiles natively in the Mercator projection, the Mercator tiles may be created as they are requested by the map client. On-the-fly reprojection of Mercator tiles is fast and accurate when using the table-based reprojection method discussed above. An efficient tile cache for the Mercator tiles will ensure that the most commonly requested tiles are available with no additional delay over native Geodetic tiles. If the distortion of the Geodetic projection is a problem, and the primary clients are tile-based, then server-side Mercator reprojection is useful.

On the other hand, if distortion is a problem, but the map client uses continuous scales or a WMS interface, client-side reprojection may be a better option. A client that performs its own reprojection will be able to use any dataset in the Geodetic projection, preserving interoperability. After all the data is retrieved and combined, the new aggregate image may be reprojected once.

If the client performs reprojection, it may choose the target projection best suited for its users. For example, the appropriate UTM projection may be used when the map scale warrants it. Having a client project back to UTM when necessary removes problems of tiles not aligning if stored natively in UTM. Of course, the UTM reprojection is computationally costly and may overwhelm the client system.

An alternative solution would be to use a simpler reprojection method, which reduces distortion with a simple linear transformation. This simple reprojection method works by scaling a map image to ensure the ground distance for the width and height of a pixel is equal. The scaled image is both equal-area and conformal, though neither is accurate enough for cartographic applications. Code for performing the bounding box modification is shown in Listing 10.1.

This simple reprojection method can be encoded in the request where the client modifies the geographic bounds to match the latitude and longitude range given by the above formulae. The benefit of encoding the scaling in the request is that it offloads the image manipulation to the server. If the client displays maps using tiles natively, rather than a single composite image created by the server, using this reprojection method becomes more complicated. Using the above formulae on each image tile individually will cause tile alignment issues, much like with the affine transform reprojection. Instead, the formulae should be calculated for a single average latitude for all tiles. Using the same central latitude will ensure the same scaling is performed on each tile. However, further complications arrise as the map view moves north or south and the average latitude for the screen changes. Refreshes of all tiles on the screen will need to occur on a regular basis to ensure the scaling performs adequately.

10.6 Conclusion

Dealing with map projections, datums, and coordinate systems is complicated. This chapter has tried to explain those portions of these topics necessary to properly build a tiled-map system. Our primary focus has been on techniques that provide proper balance to the tiled-mapping system. Tabular point-wise reprojection provides a good balance between accuracy and computations cost. The Geodetic projection provides simplicity and interoperability while allowing client-side adjustments to reduce its distortion.

Interoperability is a primary concern for most tiled-mapping systems. In general, they are not going to be large systems like Google Maps, Yahoo Maps, or Bing Maps which have no interest in providing data outside of their clients. These systems have the core mission to provide street level mapping and, therefore, an impetus to use the Mercator projection. For tile-mapping systems designed to be used in generic clients using Web services, using a non-Geodetic projection would be a significant impediment to general acceptance. Additionally, with the tabular point-wise reprojection algorithm discussed in this chapter, on-the-fly reprojection from Geodetic tiles to Mercator tiles is certainly feasible. There really is no best projection for all purposes, but the simplicity of the Geodetic projection for computer-based mapping makes it the best choice from an interoperability perspective.

Listing 10.1 Modify a bounding box so that the ground distance for the width and height of a pixel is equal.

```
1  // this method adjust the input bounding box so it will have the same aspect
       ratio as width to height of map display
2  // this is done by adding (never subtracting) area to the bounding box
3  public static BoundingBox adjustBoundingBoxToDimension(BoundingBox bb, int
       width, int height) {
4      BoundingBox adjustedBoundingBox = new BoundingBox(0.0, 0.0, 0.0, 0.0);
5
6      // determine how many longitudinal degrees must be covered for the given
           latitudinal range
7      double averageLatitude = (((bb.maxY + bb.minY) / 2.0) / 360.0) * (2 * Math.
           PI);
8
9      //WGS84
10     double kmInALatitudeDegree = Math.abs(111.13292 - 0.55982 * Math.cos(2.0 *
           averageLatitude) + 0.001175 * Math.cos(4.0 * averageLatitude) -
           0.0000023 * Math.cos(6.0 * averageLatitude));
11
12     double kmInALongitudeDegree = Math.abs(111.41284 * Math.cos(averageLatitude
           ) - 0.0935 * Math.cos(3.0 * averageLatitude) + 0.000118 * Math.cos(5.0
            * averageLatitude));
13
14     // how many degrees of longitude must be covered so that an equal distance
           is covered as is covered by the latitude range
15     double mapAspectRatio = ((double)(width)) / ((double)(height));
16
17
18     double kmCoveredInLatitude = Math.abs(kmInALatitudeDegree * (bb.maxY - bb.
           minY));
19     double kmCoveredInLongitude = Math.abs(kmInALongitudeDegree * (bb.maxX - bb
           .minX));
20
21     double groundDistanceAspectRatio = kmCoveredInLongitude /
           kmCoveredInLatitude;
22
23     if (groundDistanceAspectRatio > mapAspectRatio) { // latitude range must be
            expanded
24         double adjustedLatitudeRange = ((1 / mapAspectRatio) *
               kmCoveredInLongitude) / kmInALatitudeDegree;
25
26         adjustedBoundingBox.minX = bb.minX;
27         adjustedBoundingBox.maxX = bb.maxX;
28         adjustedBoundingBox.minY = (float)(bb.minY - Math.abs((
               adjustedLatitudeRange - Math.abs(bb.maxY - bb.minY)) / 2.0));
29         adjustedBoundingBox.maxY = (float)(bb.maxY + Math.abs((
               adjustedLatitudeRange - Math.abs(bb.maxY - bb.minY)) / 2.0));
30     }
31     else { // longitude range must be expanded
32         double adjustedLongitudeRange = (mapAspectRatio * kmCoveredInLatitude)
               / kmInALongitudeDegree;
33         adjustedBoundingBox.minY = bb.minY;
34         adjustedBoundingBox.maxY = bb.maxY;
35         adjustedBoundingBox.minX = (float)(bb.minX - Math.abs((
               adjustedLongitudeRange - Math.abs(bb.maxX - bb.minX)) / 2.0));
36         adjustedBoundingBox.maxX = (float)(bb.maxX + Math.abs((
               adjustedLongitudeRange - Math.abs(bb.maxX - bb.minX)) / 2.0));
37     }
38
39     return adjustedBoundingBox;
40 }
```

Listing 10.2 Python code to convert a geodetic point to UTM.

```
from math import *

# longitude and latitude should be in degrees
# we will convert to radians in the code
# longitude should be between [-180, 180)
# latitude should be between [-80, 84)
def GeodeticToUTM(lon, lat):

    # a and b are WGS84 ellipsoid constants
    a = 6378137.0
    b = 6356752.3

    # k0 is a scaling factor used to reduce average distortion
    k0 = 0.9996

    # the next section of statements calculates the zone
    # there are some special exceptions
    if (lat >= 56 and lat < 64 and lon >= 3 and lon < 12):
        zone = 32

    # this is special zones for Svalbard
    elif (lat >= 72 and lat < 84):
        if (lon >= 0 and lon < 9):
            zone = 31
        elif (lon >= 9 and lon < 21):
            zone = 33
        elif (lon >= 21 and lon < 33):
            zone = 35
        elif (lon >= 33 and lon < 42):
            zone = 37

    # default formula for calculating zone
    else:
        zone = int((lon + 180) / 6) + 1

    lonRad = radians(lon)
    latRad = radians(lat)

    # calculate the central longitude and convert to radians
    centralLon = (zone - 1) * 6 - 180 + 3
    centralLonRad = radians(centralLon)

    # do calculations to convert to UTM
    e = sqrt(1- (b*b)/(a*a))
    e2 = e*e
    ePrime2 = (e2) / (1- (e2))

    N = a / sqrt(1 - e2 * pow(sin(latRad), 2))
    T = pow(tan(latRad), 2)
    C = ePrime2 * pow(cos(latRad), 2)
    A = cos(latRad) * (lonRad - centralLonRad)

    M = a * (( 1-e2 / 4 - 3 * pow(e2,2) / 64 - 5 * pow(e2, 3) / 256) * latRad -
          (3 * e2 / 8 + 3 * pow(e2, 2) / 32 + 45 * pow(e2, 3) / 1024) * sin(2*
         latRad) + (15 * pow(e2, 2) / 256 + 45 * pow(e2, 3)/1024) * sin(4*
         latRad) - (35*pow(e2, 3) / 3072) * sin(6*latRad))

    easting = k0 * N * (A  + (1-T+C)*pow(A,3)/6+(5-18*pow(T,3)+72*C-58*ePrime2)
          * pow(A,5)/120) + 500000.0
    northing = k0 * (M +N * tan(latRad) * (pow(A,2) / 2+ (5-T+9*C+4*pow(C,2)) *
          pow(A,4) / 24 + (61 - 58 * T + pow(T,2) + 600 * C - 330 * ePrime2) *
         pow(A,6)/720))

    hemisphere = 'N'
    # if southern hemisphere then add the false northing
    if (lat < 0):
```

```
61          northing = northing + 10000000
62          hemisphere = 'S'
63
64
65      return (easting, northing, zone, hemisphere)
66
67
68  for i in xrange(100000000):
69      GeodeticToUTM(-90, 30)
```

References

1. Goldberg, D., Gott, J.: Flexion and skewness in map projections of the Earth. Cartographica: The International Journal for Geographic Information and Geovisualization **42**(4), 297–318 (2007)
2. Iliffe, J.: Datums and Map Projections for remote sensing, GIS, and surveying. Whittles Publishing (2000)
3. Jain, S., Barclay, T.: Adding the EPSG: 4326 Geographic Longitude-Latitude Projection to TerraServer (2003)
4. Mesick, H., Ioup, E., Sample, J.: A Faster Technique for the Transformation of Universal Transverse Mercator Projected. Tech. rep., Naval Research Laboratory (2004)
5. Robinson, A.: A new map projection: Its development and characteristics. International Yearbook of Cartography **14**(1974), 145–155 (1974)
6. Robinson, A., Morrison, J., Muehrcke, P., Kimerling, A., Guptill, S.: Elements of Cartography, 6th edn. John Willey & Sons (1995)
7. Snyder, J.: Map projections: a working manual. USGPO (1987)