Tile based games

Piotr Fulmański

Wydział Matematyki i Informatyki,
Uniwersytet Łódzki, Polska

7 października 2016

Table of contents

Aim of this lecture

In this lecture we discusse how axonometric projections may be used in computer graphics, multimedia applications and computer games. We compare the **linear perspective** to the **axonometric projection** or **parallel perspective**, lists the major properties and tackles some implementation details. Our focus is on the **isometric** and **dimetric projections**, the most widely used varieties of the axonometric projection.

Graphical projection

Definition

Graphical projection is a protocol by which an image of a three-dimensional object is projected onto a planar surface (without the aid of mathematical calculation) used in technical drawing.

There are two types of graphical projection, categories each with its own protocol

- perspective projection,
- parallel projection.

Perspective projection

Definition

Perspective (from Latin *perspicere*, to see through (pl. *przenikać wzrokiem; przyglądać się; badać*)) in the graphic arts, such as drawing, is an approximate representation, on a flat surface (such as paper), of an image as it is seen by the eye.

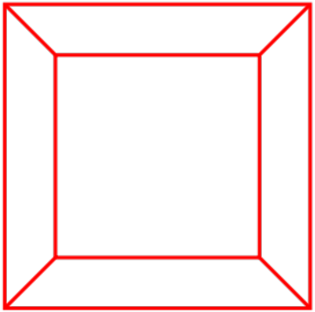
Perspective projection

Perspective projection is a type of projection where **three dimensional objects are not projected along parallel lines, but along lines emerging from a single point**. This means that

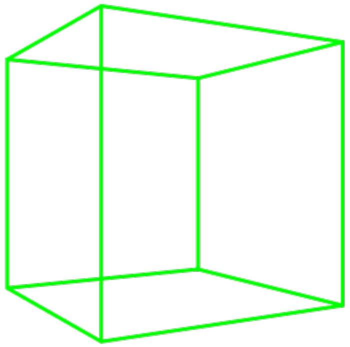
- distant objects appear smaller than nearer objects;
- lines which are parallel in nature appear to intersect in the projected image.

For example if railways are pictured with perspective projection, they appear to converge towards a single point, called **vanishing point**. Photographic lenses and the human eye work in the same way, therefore perspective projection looks most realistic.Perspective projection is usually categorized into one-point, two-point and three-point perspective, depending on the orientation of the projection plane towards the axes of the depicted object.

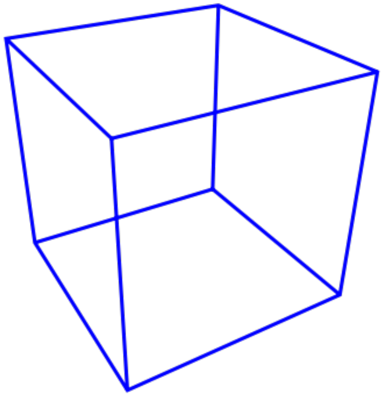
Perspective projection



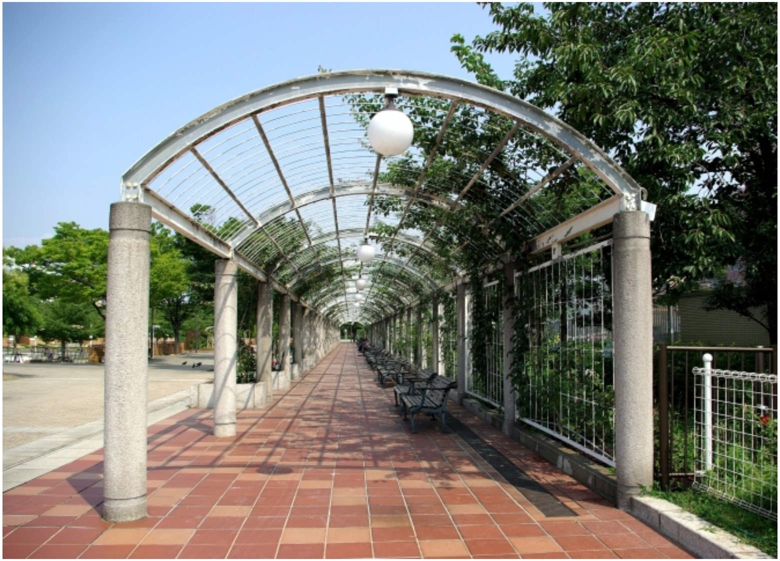
Perspective projection



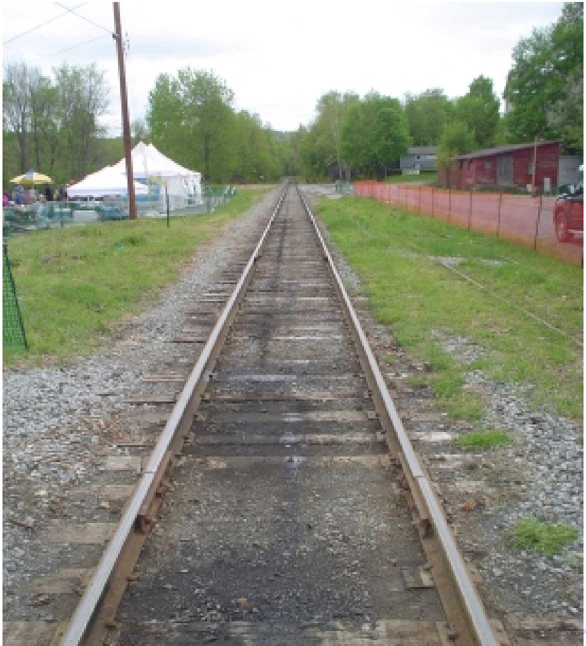
Perspective projection



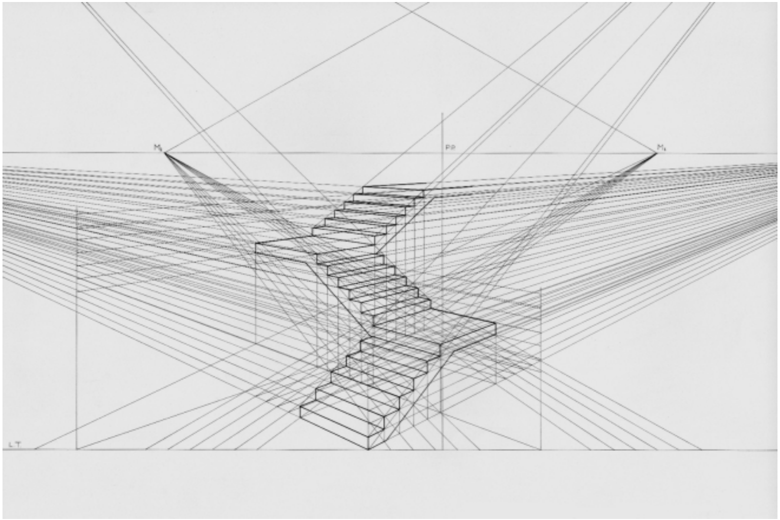
Perspective projection



Perspective projection



Perspective projection



Parallel projection

In the Western world, we are accustomed to the linear perspective but another perspective had developed in oriental art. The **Chinese perspective** was an intrinsic part of the classical scroll painting. A typical Chinese scroll painting had a size of approximately 40 centimetres high by several meters wide. One views the painting by unrolling it on a table in segments of about 60 centimetres wide. We can say, that the Chinese scroll paintings show a development in time – a form of „narrative art”, in contrast to the paintings that were made in Europe at the time, which show a „situation” rather than a development.

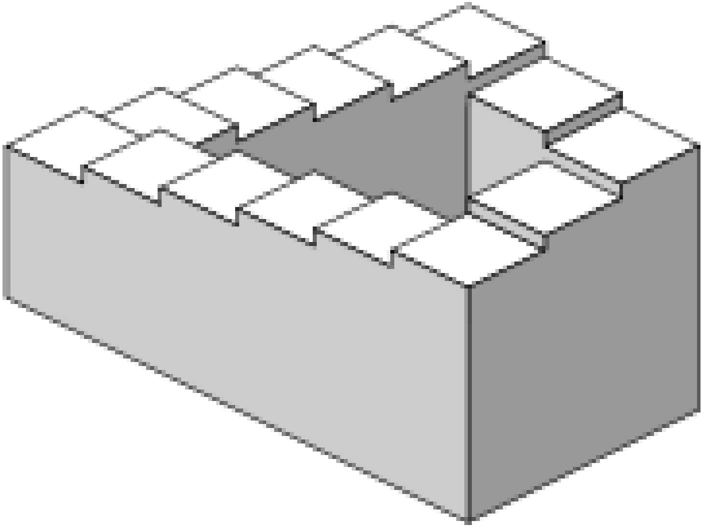
Parallel projection

Along the River During Qingming Festival

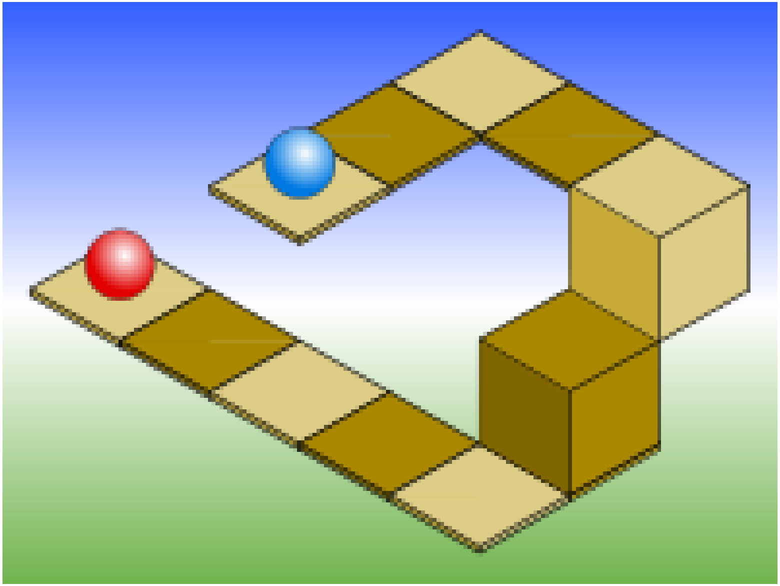
Objects drawn with parallel projection do not appear larger or smaller as they extend closer to or away from the viewer. While advantageous for technical drawings, where measurements must be taken directly from the image, the result is a perceived distortion, since unlike perspective projection, this is not how our eyes or photography normally work. It also can easily result in situations where depth and altitude are difficult to gauge.

Parallel projection

Problems with parallel projection



Problems with parallel projection



One of the most important type of parallel projection is an **axonometric projection**. Axonometric projection is a type of parallel projection used to create a pictorial drawing of an object, where the object is rotated along one or more of its axes relative to the plane of projection.

- Parallel projections have lines of projection that are parallel both in reality and in the projection plane.
- Axonometric projection is a type of parallel projection where the object is rotated along one or more of its axes relative to the plane of projection.
- Axonometric projection is a type of parallel projection where the plane or axis of the object depicted is not parallel to the projection plane, such that multiple sides of an object are visible in the same image.

There are three main types of axonometric projection which is a type of parallel projection:

- isometric,
- dimetric,
- and trimetric projection,

depending on the exact angle at which the view deviates from the orthogonal.

Parallel projection

Axonometric projection – isometric projection

Definition

Isometric projection: This type of projection is widely used in technical and engineering drawings. The three coordinate axes are equally foreshortened and the angle between them is 120 degree. This makes it possible for the observer to have a sense of the object's real proportions.

Definition

Dimetric projection: In this projection only two of the axes are equally foreshortened. The scale of the third direction is determined separately. This projection is most often used in videogames and pixel art.

Parallel projection

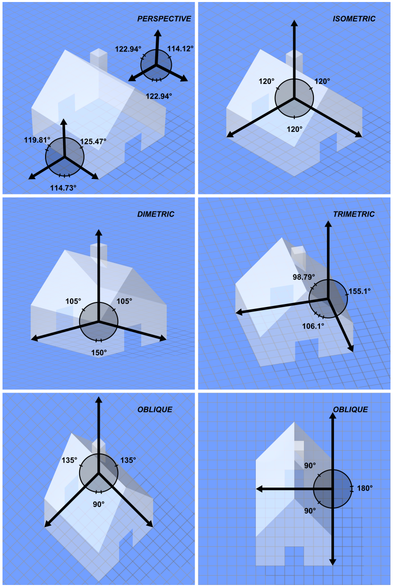
Axonometric projection – trimetric projection

Definition

Trimetric projection: Here the direction of observing is such that all of the axes appear unequally foreshortened.

Parallel projection

Axonometric projection



Parallel projection

Axonometric projection – important property

A more important property of axonometry is its **fixed relation between sizes of objects in world space and those on projected space, independent of the positions of the objects in projected space**. In linear perspective, objects become smaller when they move farther away; not so in axonometric perspective. This means that we can measure the size of an object of a axonometric drawing and know how big the real object is (we need to know the scale of the drawing and the properties of the projection, but nothing else), something that cannot be done with linear perspective. This leads to the name of the projection: the word *axonometry* means *measurable from the axes*.

Parallel projection

Axonometric projection – important property

The main properties of axonometric projections are:

- No vanishing points. This allows to scroll a large image below a viewport and to have the same perspective at any point.
- Lines that are parallel in the 3-dimensional space remain parallel in the 2-dimensional picture. This is in contrast with the linear perspective, where parallel lines along the z-axis in the 3-dimensional space collapse to a single vanishing point at the horizon in the 2-dimensional picture.
- Objects that are distant have the same size as objects that are close; objects do not get smaller as they move away from the viewer. If we know the scale of the axes, we can measure the size (width, height, length, depth) of an object directly from the picture, regardless of its position on the z-axis; hence the name axonometry.
- The axonometric projections are standardized for technical drawings. These standards are optimized for ease of use versus visual realism.

In practice (in computer graphics) axonometry, is a drawing technique where the orthogonal $x-$, $y-$ and $z-$axes of the (3-dimensional) world space are projected to (non-orthogonal) axes on paper. In the projection, the $y-$axis usually remains the vertical axis, the $z-$axis is skewed and the $x-$axis may either be horizontal, or be skewed as well.
Although there are countless possible axonometric projections, only two are standardized for technical drawings which are discussed in the following.

Parallel projection

Axonometric projection – practice

When we want to use axonomertic projection two problems rises:

1. What are common (or well-proportioned) axonometric projections, and how persvasive does each look?
2. At what angles does one look at board in an axonometric projection? It is tempting to use rendered 3-D objects on an axonometric map, as sprites. To specify the position and orientation of the camera (view point) in relation to the object, we will need to know the intrinsic angles of the axonometric map that we are using.

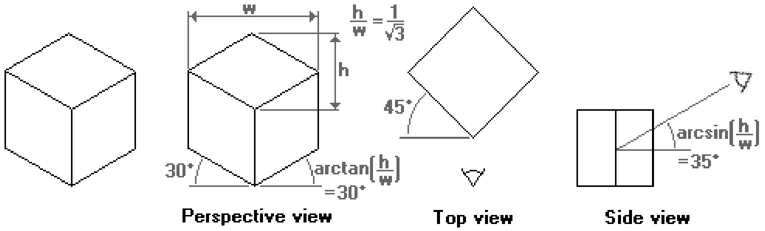
Parallel projection

Axonometric projection – practice: the isometric projection

As we know from the previous part, the isometric projection is a type of axonometric projection which is in turn a type of parallel projection. The term *isometric* comes from the Greek for *equal measure* (gr. *isos* – equal), reflecting that the scale along each axis of the projection is the same (unlike some other forms of graphical projection). It means that in an isometric projection, the $x-$, $y-$ and $z-$axes have the same metric: a unit along the $x-$axis is equally long along the $y-$ and $z-$axes. In other words, if we render a wire frame cube, all edges in the 2-dimensional picture are equally long. Another property of the isometric projection is that the projected wire frame cube is also symmetric. All sides of the projected cube are rhombuses.

Isometric projection as a standard

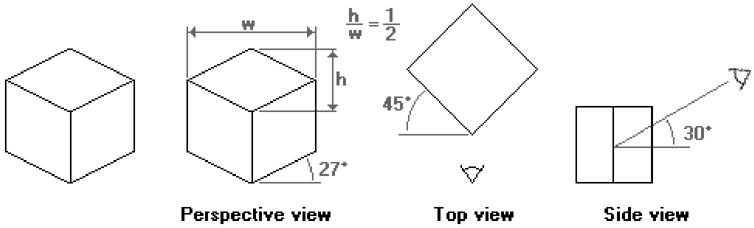
The international standard ISO 5456-3 (Technical drawings – Projection methods – Part 3: Axonometric representations; PN-EN ISO 5456-3:2002 Rysunek techniczny – Metody rzutowania – Część 3: Przedstawianie aksonometryczne) describes an isometric projection that is symmetric with regards to the vertical axis; the angle between the $x-$ and $y-$axes, and between the $z-$ and $y-$axes, is 60 degrees. The projection shows three sides of a cube, and the surfaces of each side are equal. The 30 angle between the $x-$ and $z-$axes and the „horizon" is convenient for technical drawings, because the sine of 30 is $\frac{1}{2}$.



w
$\frac{h}{w}=\frac{1}{\sqrt{3}}$
h
30°
$\text{arctan}\left(\frac{h}{w}\right)$
=30°
45°
$\text{arcsin}\left(\frac{h}{w}\right)$
=35°
Perspective view
Top view
Side view

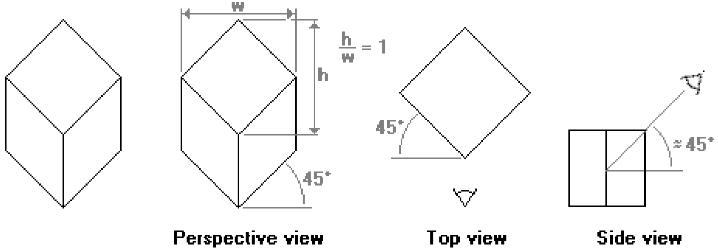
Isometric projection in computer games

Computer games with isometric maps are often tile-based. To make tiles match, the game designer must take into account how diagonal lines are plotted in discrete steps. As it turns out, a line at 30 degrees (sine is 0.5) produces steps that are too irregular. A line at an angle where the tangent is 0.5 does have a nice regular pattern: two steps to the right, one step up. Thus, the isometric projection used by most games tilt the $x-$ and $z-$axes with approximately 27 degrees (the exact angle is $\operatorname{arc}\operatorname{tg}(0.5)$). Because the tangent of the angle of the rhombus is 0.5, the rhombus is twice as wide as it is high. This is why many sources mention a 1:2 scale for isometric perspectives.



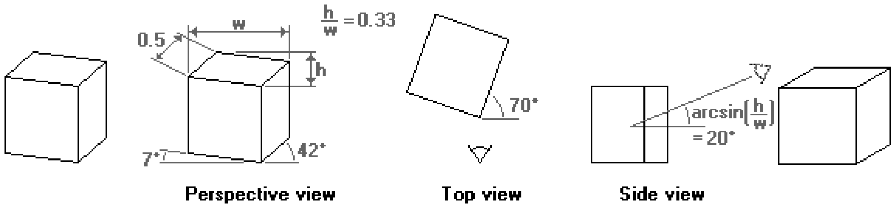
Isometric projection in military

The two isometric projections mentioned above „skew” all faces of the example cube. In applications where the most important faces are the horizontal ones, for example in applications where maps or floor plans are paramount, another isometric projection is common. It is called the „military” projection, probably because of its origin or use. In the military projection, the angles of the $x-$ and $z-$axes are at 45, meaning that the angle between the $x-$axis and the $z-$axis is 90. That is, the $x-z$ plane is not skewed. It is rotated over 45, though.

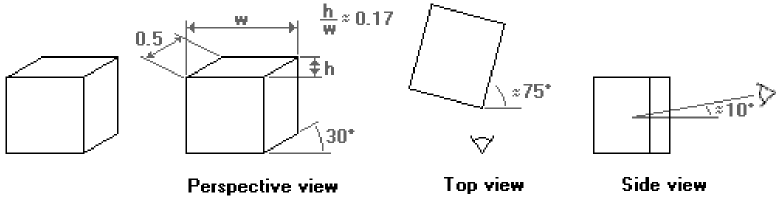


The dimetric projection (iso case)

In the dimetric projection, one of the three axes has a different scale than the other two. In practice, the scaled axis is the $z-$axis and, hence, a cube drawn in a dimetric projection is not a symmetrical graphic (like in the isometric projection).

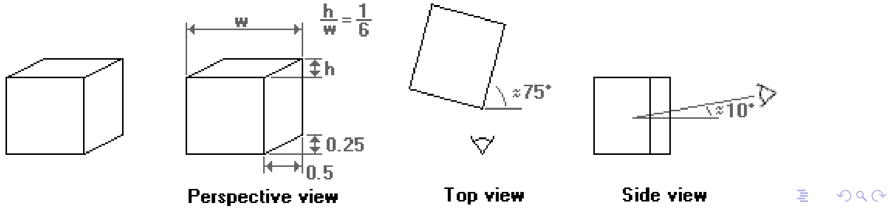


The dimetric projection (chinese case)



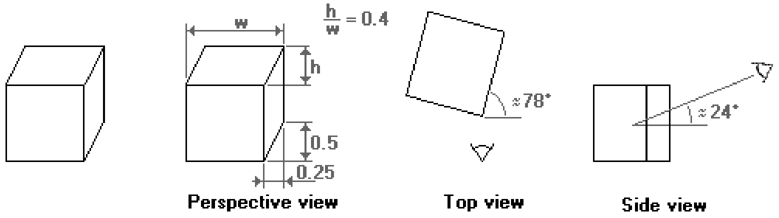
Dimetric projection in computer games

As was the case with the isometric projection, in computer graphics some angles are preferable to others. The first dimetric projection we can use for (tiled) computer graphics is very similar to the projection of Chinese scroll paintings. The difference is the scale of the z-axis, and the angle that it makes with the $x-$axis. To start with the angle, the z-axis is slanted with approximately 27 degrees (the exact angle is $\operatorname{arc\,tg}(0.5)$). This is the same angle as the isometric projection for computer graphics uses. The scale is such that the width of the side view of a cube, when measured along the $x-$axis, is half of the width of the front face. The key phrase in the previous sentence is „when measured along the $x-$axis". In the two former projections, the scale factor applied to distances measured along the $z-$axis.



Dimetric projection in platform games

The above projection gives a perspective that is viewed mostly from the side. I might be useful to add some depth to a side-scrolling (or „platform”) game. For board-like games, a perspective that is viewed from a greater height is more appropriate. The second proposed dimetric projection for games serves this end. Again, note that the perspective of the two projections suggested above is distorted. The angles in the top and side views are really approximate. For example, in the second projection the angle at which one looks from above at the scene is given as 24 degrees. However, using an angle of 30 degrees may actually look better. In addition, a 30-degree angle lets us use the same objects for both the dimetric and the isometric projections for games.



Conversion from 3D to 2D: Isometric

Converting from space coordinates (x, y, z) to a pixel coordinate (x', y') in the projection requires only trivial geometry (we consider left-handed orientation).

$$\begin{aligned} x' &= (x - z)\cos(30) \\ y' &= y + (x + z)\sin(30) \end{aligned}$$

Conversion from 3D to 2D: Isometric (computer games)

$$\begin{aligned} x' &= (x - z) \\ y' &= y + \tfrac{x+z}{2} \end{aligned}$$

Conversion from 3D to 2D: Dimetric (ISO)

$$
\begin{aligned}
x' &= x\cos(7) + \frac{z\cos(42)}{2} \\
y' &= y + \frac{z\sin(42)}{2} - x\sin(7)
\end{aligned}
$$

Conversion from 3D to 2D: Dimetric (computer games: side view)

$$\begin{aligned} x' &= x + \frac{z}{2} \\ y' &= y + \frac{z}{4} \end{aligned}$$

Conversion from 3D to 2D: Dimetric (computer games: top view)

$$\begin{aligned} x' &= x + \frac{z}{4} \\ y' &= y + \frac{z}{2} \end{aligned}$$

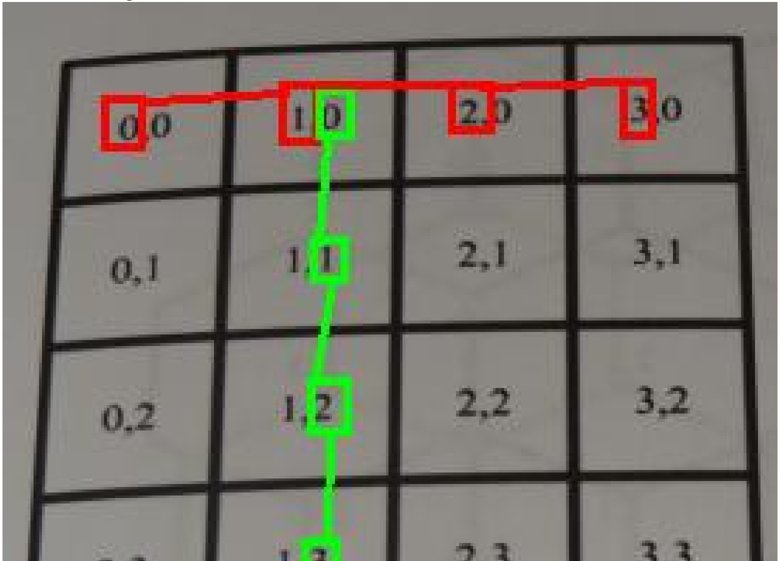
A tile-based game is a game that uses tiles as one of the fundamental elements of play. Traditional tile-based games use small tiles as playing pieces for gambling or entertainment games.

Three types of tiles

- **Square.** A tile-based game often uses square ar rectangular sections of the screen to represent the individual tile. Typically, whatever occupies the tile is contained wholly within it and does not extend beyond the borders of the square are rectangle. This makes rendering portions of rectangular maps really easy, as you can redraw a portion of the map without worrying too much about the other portions.
- **Rhombus.** Unlike rectangular maps, the tiles in an isometric grid almost always have a graphic that exceeds the boundaries of the tile. This makes the rendering order of the tiles important.
- **Hexagon.** A tile-based game that uses a hexagonal map consists of hexagons. Like rectangular maps, the tiles in a hex grid usually entirely contain the objects within them, so rendering updates is pretty simple but still not as trivial as rectangular maps.

Isometric and hexagonal maps. How maps are constructed – square case

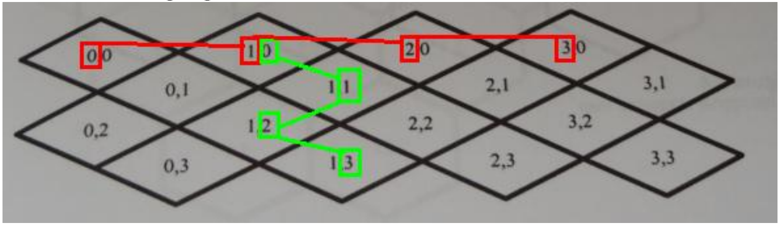
In a rectangular map, life is easy.



Isometric and hexagonal maps. How maps are constructed – rhombus (isometric) case

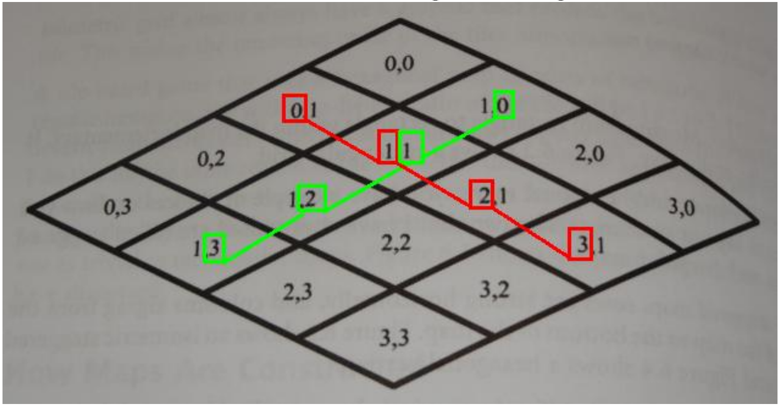
In this case we have a choice between staggered map and diamond map.

- In a staggered map, rows are strung horizontally,
  and columns zigzag from the top of the map to the bottom of the map.



Isometric and hexagonal maps. How maps are constructed – rhombus (isometric) case

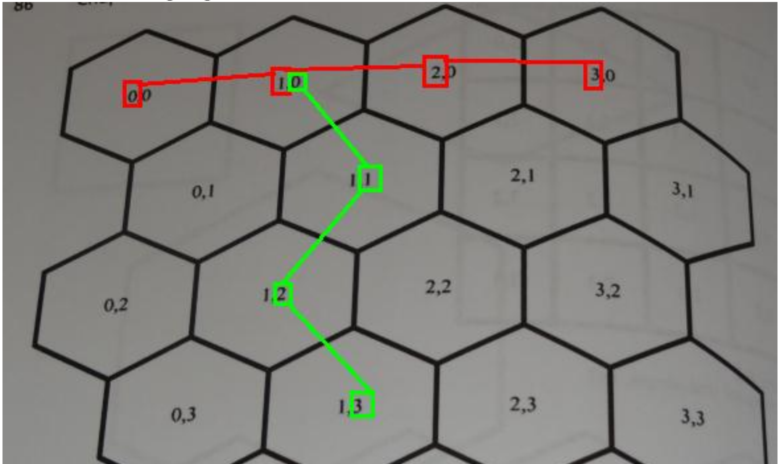
- In a diamond map, rows and columns go on a diagonal.



Isometric and hexagonal maps. How maps are constructed – hexagon case

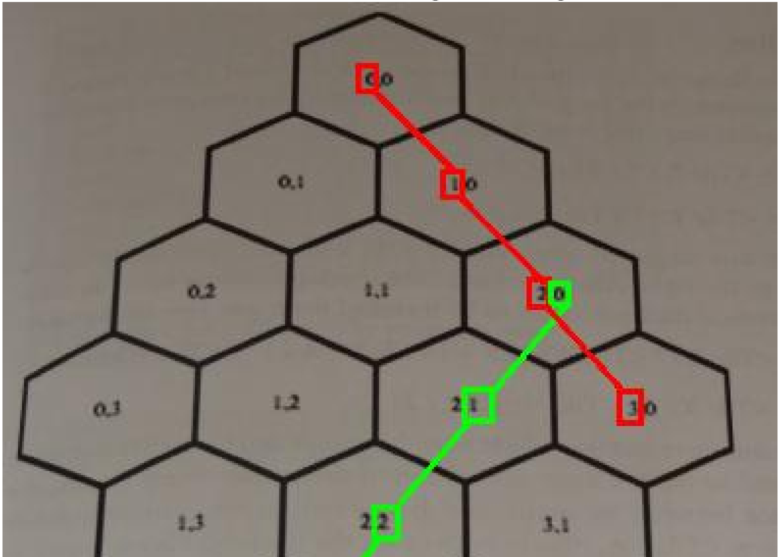
In this case, like for rhombus case, we have a choice between staggered map and diamond map.

- In a staggered map, rows are strung horizontally, and columns zigzag from the top of the map to the bottom of the map.



Isometric and hexagonal maps. How maps are constructed – hexagon case

- In a diamond map, rows and columns go on a diagonal.

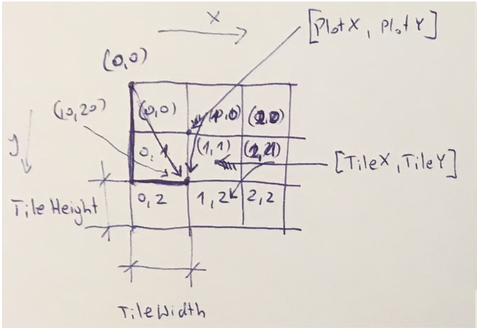


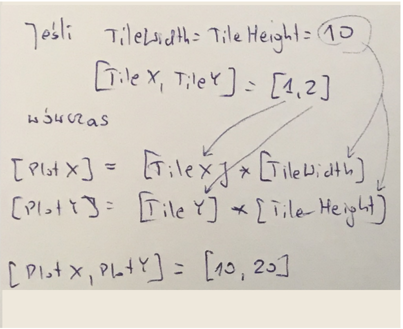
Isometric and hexagonal maps. Ploting tales – rectangular case

One of the main things to deal with when rendering a tile-based game is taking a map location somewhere on the grid and determining where on screen to render it. For a rectangular map, this is quite simple

$$[PlotX] = [TileX] * [TileWidth]$$
$$[PlotY] = [TileY] * [TileHeight]$$

Isometric and hexagonal maps. Ploting tales – rectangular case




Jeśli TileWidth = Tile Height = 10

[Tile X, Tile Y] = [1,2]

wówczas

[Plot X] = [Tile X] * [TileWidth]

[Plot Y] = [Tile Y] * [Tile Height]

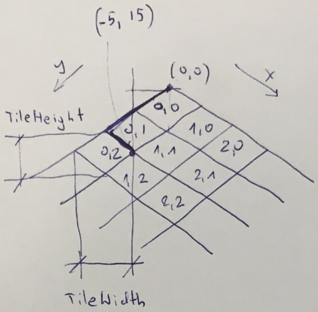
[Plot X, Plot Y] = [10, 20]

Isometric and hexagonal maps. Ploting tales – diamond case

In a diamond map the x values of tiles increase downward and to the right, whereas the y values increase downward and to the left.

$$[PlotX] = ([TileX] - [TileY]) * ([TileWidth]/2)$$
$$[PlotY] = ([TileX] + [TileY]) * ([TileHeight]/2)$$

Isometric and hexagonal maps. Ploting tales – diamond case for rhombus



Jeśli: TileWidth = TileHeight = 10

[TileX, TileY] = [1, 2]

wówczas

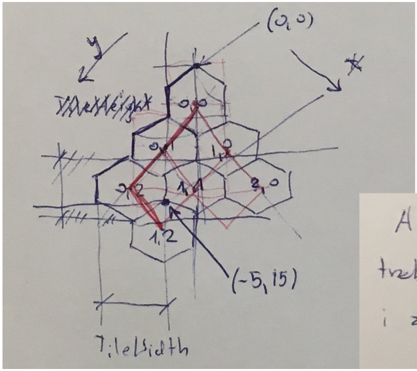
[PlotX] = [TileX - TileY] * ([TileWidth]/2)

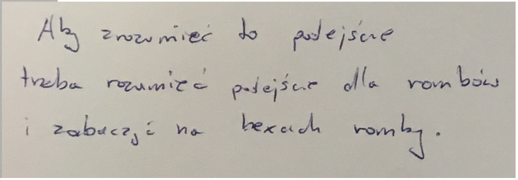
[PlotY] = [TileX + TileY] * ([TileHeight]/2)

[PlotX, ~~PlotY~~] = [1-2] * ([10]/2) = -1 * 5 = -5

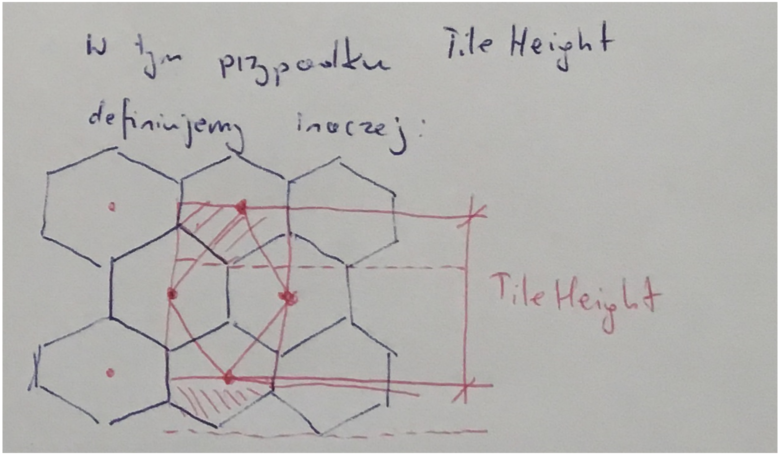
PlotY = [1+2] * ([10]/2) = 3 · 5 = 15

Isometric and hexagonal maps. Ploting tales – diamond case for hex

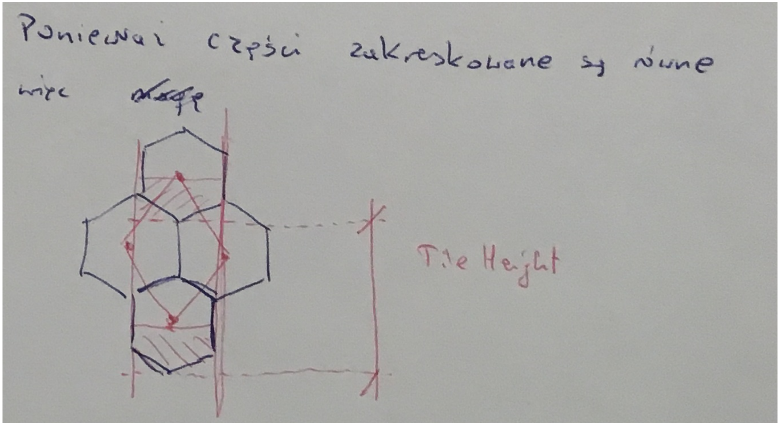



Aby zrozumieć to podejście trzeba rozumieć podejście dla rombów i zobaczyć na hexach romby.

Isometric and hexagonal maps. Ploting tales – diamond case for hex

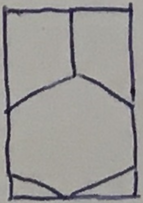


Isometric and hexagonal maps. Ploting tales – diamond case for hex

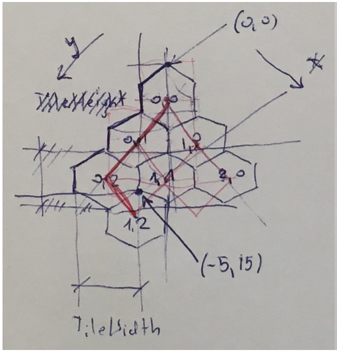

Ponieważ części zakreskowane są równe
więc

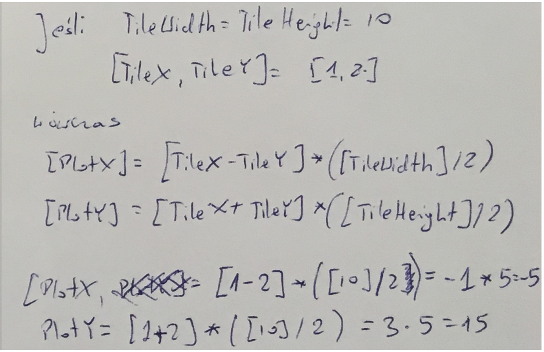
Isometric and hexagonal maps. Ploting tales – diamond case for hex

Stąd mamy, że wysokości
hexa określa się dla hexa
postaci



Isometric and hexagonal maps. Ploting tales – diamond case for hex




Jeśli: TileWidth = TileHeight = 10

[TileX, TileY] = [1, 2]

wówczas

[PlotX] = [TileX - TileY] * ([TileWidth]/2)

[PlotY] = [TileX + TileY] * ([TileHeight]/2)

[PlotX, ~~PlotY~~ = [1-2] * ([10]/2) = -1 * 5 = -5

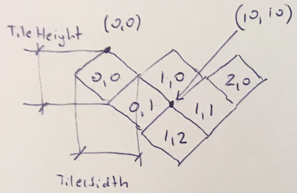
PlotY = [1+2] * ([10]/2) = 3 · 5 = 15

Isometric and hexagonal maps. Ploting tales – staggered maps (rhombusa and hex)

In a staggered maps (rhombusa and hex), we have that every other row is offset half a tile to the right. Also, there is a 50% overlap between rows, so only one half of the vertical distance needs to be travelled from one row to the next.

$$[PlotX] = [TileX] * [TileWidth] + ([TileY]\%2) * ([TileWidth]/2)$$
$$[PlotY] = [TileY] * ([TileHeight]/2)$$

Isometric and hexagonal maps. Ploting tales – staggered maps for rhombus



Jeśli TileWidth = TileHeight = 10

[TileX, TileY] = [1,2]

to wówczas

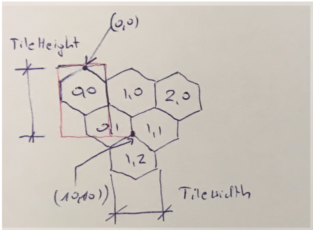
[PlotX] = [TileX] * [TileWidth] + ([TileY] % 2) * ([TileWidth]/2)

[PlotY] = [TileY] * ([TileHeight]/2)

[PlotX] = 1 * 10 + (2 % 2) * (10/2) = 10 + 0 * 5 = 10

[PlotY] = 2 * (10/2) = 2 * 5 = 10

Isometric and hexagonal maps. Ploting tales – staggered maps for hex



Jeśli TileWidth=TileHeight=10

[TileX, TileY]=[1,2]

to wówczas

[PlotX]=[TileX]*[TileWidth]+([TileY]%2)*([TileWidth]/2)

[PlotY]=[TileY]*([TileHeight]/2)

[PlotX]= 1*10+(2%2)*(10/2)=10+0*5 = 10

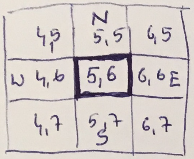
[PlotY]= 2*(10/2) = 2*5 =10

Isometric and hexagonal maps. Walking tales – rectangular map

Another important task in any tile-based game is moving from one tile location to another. In short, we must add some numbers to present x and y to move in a selected direction. For rectangular map it is very simple.

Direction	X **difference**	Y **difference**
North	0	-1
Northeast	1	-1
East	1	0
Southeast	1	1
South	0	1
Southwest	-1	1
West	-1	0
Northwest	-1	-1

Isometric and hexagonal maps. Walking tales – rectangular map: example



Jeśli polem odniesienia jest pole (5, 6) to

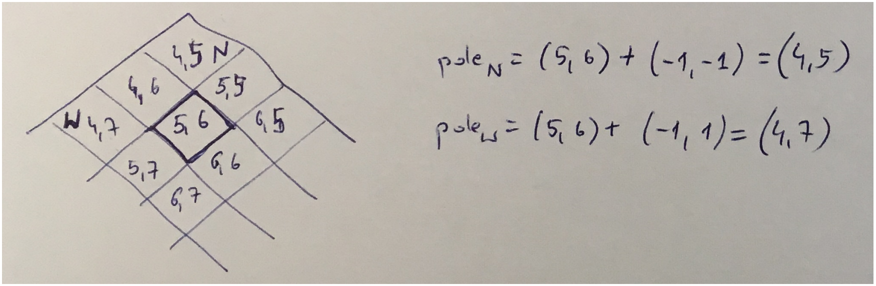
$pole_N = (5,6) + (x_{diff}, y_{diff}) = (5,6) + (0,-1) = (5,5)$

$pole_W = (5,6) + (-1,0) = (4,6)$

Isometric and hexagonal maps. Walking tales – diamond map

Direction	X difference	Y difference
North	-1	-1
Northeast	0	-1
East	1	-1
Southeast	1	0
South	1	1
Southwest	0	1
West	-1	1
Northwest	-1	0

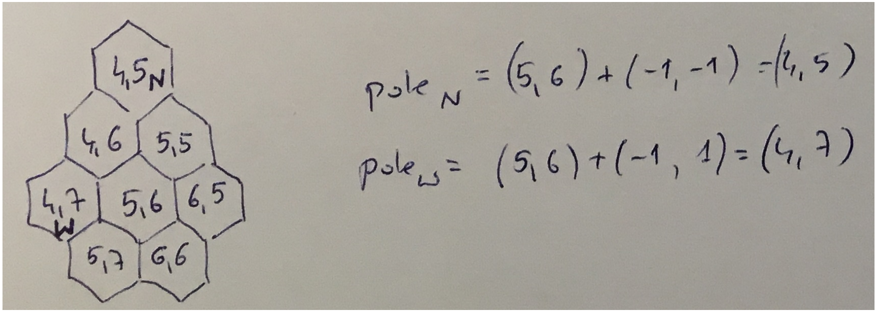
Isometric and hexagonal maps. Walking tales – diamond map: rhombus case



$$\text{pole}_N = (5,6) + (-1,-1) = (4,5)$$

$$\text{pole}_W = (5,6) + (-1,1) = (4,7)$$

Isometric and hexagonal maps. Walking tales – diamond map: hex case



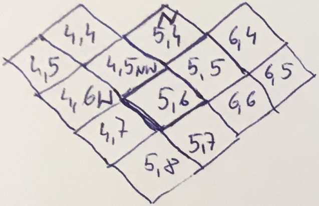
$$\text{pole}_N = (5,6) + (-1,-1) = (4,5)$$

$$\text{pole}_W = (5,6) + (-1,1) = (4,7)$$

Isometric and hexagonal maps. Walking tales – staggered map

Direction	X difference	Y difference
North	0	-2
Northeast	Y%2	-1
East	1	0
Southeast	Y%2	1
South	0	2
Southwest	Y%2-1	1
West	-1	0
Northwest	Y%2-1	-1

Isometric and hexagonal maps. Walking tales – staggered map: rhombus case



$$pole_N = (5,6) + (0,-2) = (5,4)$$

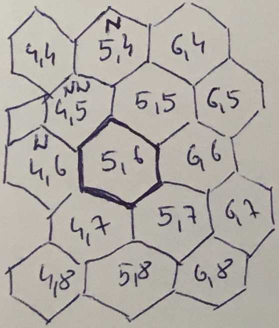
$$pole_W = (5,6) + (-1,0) = (4,6)$$

$$pole_{NW} = (5,6) + (6\%2-1, -1) =$$

$$= (5,6) + (0-1, -1) =$$

$$= (5,6) + (-1,-1) = (4,5)$$

Isometric and hexagonal maps. Walking tales – staggered map: hex case



$$pole_N = (5,6) + (0,-2) = (5,4)$$

$$pole_W = (5,6) + (-1,0) = (4,6)$$

$$pole_{NW} = (5,6) + (0-1,-1) = (4,5)$$

Isometric and hexagonal maps. Mouse mapping tales – rectangular case

$$[\mathit{TileX}] = ([\mathit{MouseX}] - [\mathit{Tile}(0,0)\mathit{ScreenX}])/[\mathit{TileWidth}]$$
$$[\mathit{TileY}] = ([\mathit{MouseY}] - [\mathit{Tile}(0,0)\mathit{ScreenY}])/[\mathit{TileHeight}]$$

Isometric and hexagonal maps. Mouse mapping tales – rectangular case

Isometric and hexagonal maps. Mouse mapping tales – staggered and diamond case; step 1

- First, we calculate rought approximation so we want to find how many steps to the east and south we need to go (starting from (0, 0) tile).

$$[RoughX] = ([MouseX] - [Tile(0,0)ScreenX])/[TileWidth]$$
$$[RoughY] = ([MouseY] - [Tile(0,0)ScreenY])/[TileHeight]$$

Isometric and hexagonal maps. Mouse mapping tales – staggered and diamond case; step 1

Isometric and hexagonal maps. Mouse mapping tales – staggered and diamond case; step 2

- Next we need to know where within the $(RoughX, RoughY)$ area the pointing device is.

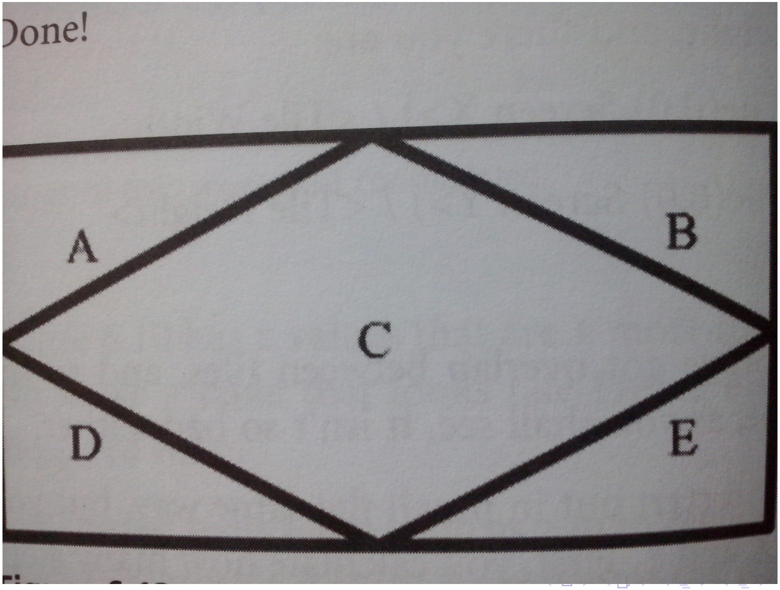
$$[FineX] = ([MouseX] - [Tile(O, O)ScreenX])\%[TileWidth]$$
$$[FineY] = ([MouseY] - [Tile(O, O)ScreenY])\%[TileHeight]$$

Isometric and hexagonal maps. Mouse mapping tales – staggered and diamond case; step 2

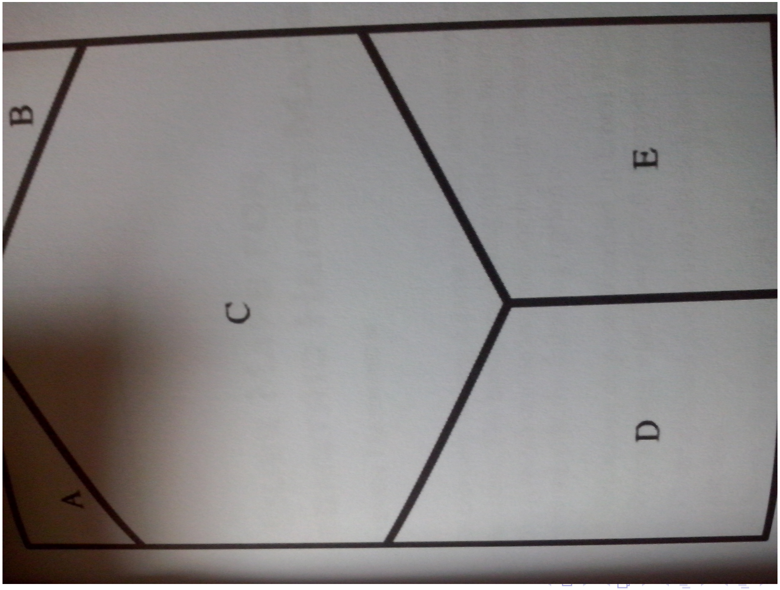
Isometric and hexagonal maps. Mouse mapping tales – staggered and diamond case; step 3

Finally we have to check the *FineX* and *FineY* coordinates into the neighborhood map. We take the *FineX* and *FineY* coordinates and use a lookup table that maps tile location to a tile in that tile neighborhood. The lookup table will be based on a map of the tiles in the area.

Isometric and hexagonal maps. Mouse mapping tales – staggered and diamond case; step 3



Isometric and hexagonal maps. Mouse mapping tales – staggered and diamond case; step 3



Isometric and hexagonal maps. Mouse mapping tales – staggered and diamond case; step 3

If the coordinate is in region A, take one step to the northwest. If in B, to the northeast. If in C, no change. If in D, one step to the southwest, and if in E, one step to the southeast.

Isometric and hexagonal maps. Mouse mapping tales – staggered and diamond case; step 3