Introducing the vector graphics engine library (vge.bas)

oog / proog.de 12/2011

The vector graphics engine library (vge.bas) has been designed to draw vector objects within a simulated world. The objects can be drawn at any size and angle. The lib has a special feature, which allows to define different levels of detail, regarding to the drawn size of the vector object.
The vge is part of the TrainSimulator program, and now it is available as a separate library.

Vector objects

A vector object is like a sheet, on which we can draw with standard graphics commands. We can place the sheet to a virtual world with just two pairs of coordinates, PointA and PointB. Moving or rotating the vector object can be done by simply set PointA and PointB to new coordinates.

Let's have a look to the first example file "example1.bas"

	program example1.bas	***comment***
	#include "vge.bas"	include the library
	Const m=1000 '1 Meter	define calculation precision for integer arithmetics (LSB = 1mm)
	Const MainViewW=640 Const MainViewH=480	define graphics screen size
	Dim As TView V	the variable V (view) contains informations, that are necessary for drawing vector models. You can define as many views, as often the world has to be rendered.
	screenres MainViewW,MainViewH,24 Color &Hf0f0f0,&H106010 Cls	switch to graphics mode

	V.Scale=1*m
	V.Debug=-1
	V.WinX=MainViewW
	V.WinY=MainViewH
		set view informations:
		Scale is 1 meter / pixel
		Debug is active (show PointA, PointB and normal vector)

	ScreenCenter 0*m, 0*m, V	set camera position for the view to 0,0
(1)	var house = VectColor(&H801515) + VectFBox(-50,-125,50,125)	build the command string to display a house and store the string in variable "house"
(2)	DrawModel 0*m, -50*m, 0*m, 50*m, house, V	draw the house on screen
	Sleep End	wait for a keypress and end the program

(1)
Program "example1.bas" starts with some global setups. Program line (1) defines the build string for the vector object "house", which is a string variable.

The build string of "house" does contain two graphics commands.

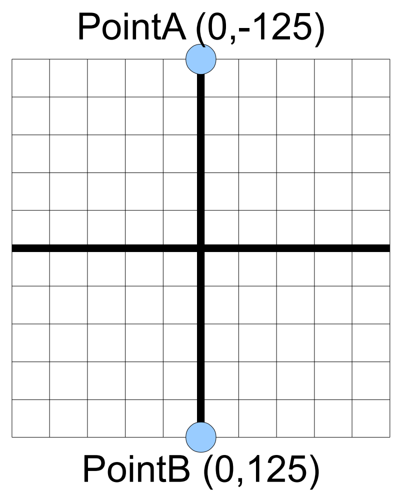
- VectColor(color) set the color to the specified RGB color value.
- VectFBox(x1,y1, x2,y2) defines a filled box (rectangle) from x1,y1 to x2,y2.

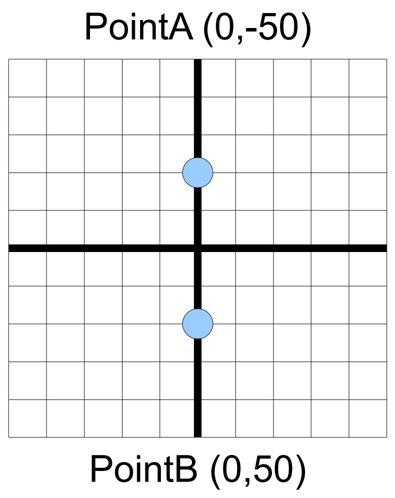
(2)
Program line (2) draws the "house" into the world. First there will be an internal check, if "house" is visible within the range of the camera. If yes, the house will be drawn.

DrawModel ax,ay, bx,by, buildstring, view, [color], [nodebug], [target]

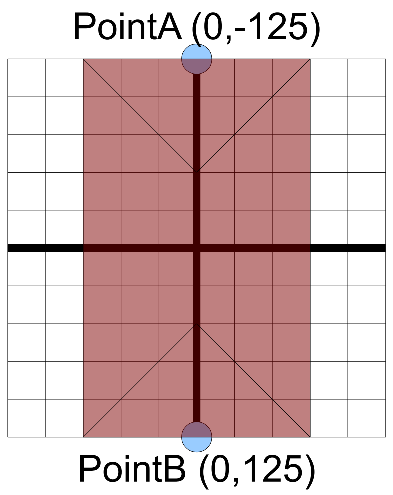
PointA and PointB is defined by ax,ay and bx,by.
buildstring = vecor model build string
view = defined view type
color = RGB value for user defined color 1
nodebug = suppress drawing of PointA and PointB if debug is active in view type.
Example for debug / nodebug: The TrainSimulator enables debug view, when in build-mode to show, where objects like houses can be moved and scaled with the mouse. Non-moveable objects like trains get the NoDebug flag, so they are drawn as usual.
target = graphics buffer where the model will be drawn to. If target is 0, the model will be drawn directly on screen.

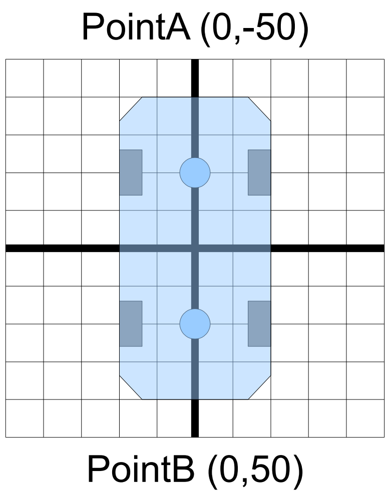
Placement Points
Usually PointA ist set to (0,-125) and PointB is set to (0,125). This definition for PointA and PointB is good for buildungs and other static objects.
For vehicles however, it's better to place PointA and PointB at the axis to make movement around turns look more natural. We use "VectVehicle" as first statement of the vector model build string, to change the placement points to vehicle mode at (0,-50) and (0,50).





This is an example for a house and a car, using the two different modes for placement points:





Run program "example2.bas" to see, what different results the choice of placement points will produce. The example defines a red box as a simple example for a building, drawn without "VectVehicle" at the left side and with "VectVehicle" at the right side.

The result may be surprising, but while the distance between PointA and PointB is the same, the right object appears larger.

More examples
Program "example3.bas" defines a racing car and an airplane. We can navigate over the map with cursor keys and zoom in and out with PageUp/PageDown keys.
The program shows animated movement of about 20 cars and the plane. The plane is scaled while moving and the cars use the "user definable color" for an individual body painting.
The models also support different levels of detail, using VectZoomIn and VectZoomOut.

Level of details
The lib supports conditional drawing, depending on the size of the vector object on screen.
We use zoom-statements to define the conditions:

- VectZoomOut(size)
- VectZoomIn(size)
- VectZoomRange(SizeOut, SizeIn)
- VectExit
- VectEndZoom

VectZoom starts an if-clause and VectEndZoom is comparable to "endif" in BASIC.

Example:

Build string	***BASIC equivalent***
VectRem("Draw simple line when zoomed out")	This is a comment. Comments are not included within the build string.
VectZoomOut(255)	If (camera is far away and object is small) Then
VectColor(1)	Set Color 1 (1 is the user definable color) Note: the RGB-value &H000001 will be used as user definable color an should not be used as a normal color
VectLine(0,-90,0,80)	Draw a line
VectExit	Exit this build string - speed up - no more to draw
VectEndZoom	EndIf

Program "example4.bas" shows the usage of VectZoomOut(ZoomLevel), VectZoomIn(ZoomLevel) and VectZoomRange(ZoomLevel Out, ZoomLevel In).
The example may be a little bit confusing, because I was playing with software-aa (anti aliasing). You can press 1 to 4 to activate it, but it cost a lot of performance and is not usable that way.
For example, aa-level 4 uses a 4 times wider and higher resolution for drawing and then calculates the color average value fom 4*4 pixel down to 1 pixel for the screen.

Press "+" and "-" to select different vector models and see, how level of details change, when you move the mouse pointer.

Note:
To make LOD selection perfectly fit together, use the same ZoomLevel values for ZoomIn and ZoomOut. The lib does a "<"-compare and a ">="-compare to make sure there will be no "missing" or "double" matches.

Example:
```
  +VectZoomOut(150)+...+VectEndZoom _
  +VectZoomIn (150)+...+VectEndZoom _        (use 150 / 150, not 150 / 149)

  +VectZoomRange(1,2)+...+VectEndZoom _
  +VectZoomRange(2,4)+...+VectEndZoom _
  +VectZoomRange(4,8)+...+VectEndZoom _
```

List of commands for the build string

VectRem(Remark)
Function VectRem(Remark As String) As String
Write a remark within the build string definition. The remark will be ignored and will not be part of the build string.

VectVehicle
Function VectVehicle As String
Set placement points into vehicle mode, PointA=(0,-50), PointB=(0,50)

VectZoomIn(ZoomLevel)
VectZoomOut(ZoomLevel)
VectZoomRange(ZoomLevel, ZoomLevel)
Function VectZoomIn(zoom As Integer) As String
Function VectZoomOut(zoom As Integer) As String
Function VectZoomRange(zoomout As Integer, zoomin As Integer) As String
Execute draw commands within the [VectZoomXXX ... VectEndZoom] block, if the condition is true.
ZoomIn: camera is nearer / object is larger as specified by "ZoomLevel".
ZoomOut: camera is more far away as specified by "ZoomLevel".
ZoomRange: camera it within the sepcified levels. The first ZoomLevel parameter has to be set to the smaller value.

[VectZoomXXX ... VectEndZoom] blocks can not be nested.

VectEndZoom
Function VectEndZoom As String
Close a conditional draw block.

VectExit
Function VectExit As String
Stop drawing this object immediately. This can be used for performance reasons. For example, if just a simple line has to be drawn for an object that is far away and very small.

Example
The build string starts with:
```
 +VectZoomOut(255) _
   +VectColor(1) _
   +VectLine(0,-90,0,80) _
   +VectExit _
 +VectEndZoom _
 + ...
```

VectColor(color)
Function VectColor(color As Integer) As String
Set the color to the specified RGB color value. The value 1 has a special meaning, 1 is a user definable color. It can be used to traw objects at different colors (see example3.bas, the color of the cars). The TrainSimulator use this color to sign the auto pilot status of the trains (green = auto pilot active, grey = inactive, red = stop).

VectLine(x1,y1, x2,y2)
Function VectLine(x1 As Integer, y1 As Integer, x2 As Integer, y2 As Integer) As String
Draw a line from x1,y2 to x2,y2 with the actual color.

VectTri(x1,y1, x2,y2, x3,y3)
VectFTri(x1,y1, x2,y2, x3,y3)
Function VectTri(x1 As Integer, y1 As Integer, x2 As Integer, y2 As Integer, x3 As Integer, y3 As Integer) As String
Function VectFTri(x1 As Integer, y1 As Integer, x2 As Integer, y2 As Integer, x3 As Integer, y3 As Integer) As String
Draw a triangle at the three defined points. The second statement draws with filled color.

VectBox(x1,y1, x2,y2)
VectFBox(x1,y1, x2,y2)
Function VectBox(x1 As Integer, y1 As Integer, x2 As Integer, y2 As Integer) As String
Function VectFBox(x1 As Integer, y1 As Integer, x2 As Integer, y2 As Integer) As String
Draw a box / rectangle from x1,y1 to x2,y2. The second statement draws with filled color.

VectCircle(x,y,r)
VectFCircle(x,y,r)
Function VectCircle(x As Integer, y As Integer, r As Integer) As String
Function VectFCircle(x As Integer, y As Integer, r As Integer) As String
Draw a circle at x,y with radius=r. The second statement draws with filled color.

ScreenCenter(x, y, v)
Sub ScreenCenter(x As Integer, y As Integer, v As TView)
Set Camera position to x,y for the specified view "v".

DrawModel(ax,ay, bx,by, buildstring, v, [color], [NoDebug], [target])
Sub DrawModel(ax As Integer, ay As Integer, bx As Integer, by As Integer, _
  mBuild As String, v As TView, col As Integer=1, _
  NoDebug As Integer=0, dwin As Any Ptr=0)
Draw vector graphics model to screen or graphics buffer. PointA = ax,ay and PointB=bx,by represent the coordinates of the vector object in the virtual world.
"v" is the specified view. "color" is the RGB value for the user definable color, which is &H000001 for the vector model build string.
NoDebug suppress the drawing of PointA and PointB when view "v" is set to debug mode.
Target secifies a graphics buffer. If target is set t 0, the vector object will be drawn on the screen.

Examples

```
var house=  _
   VectZoomOut(255) _
           +VectColor(&H800000) _
           +VectLine(0,-125,0,125) _
           +VectExit _
  +VectEndZoom _
  +VectZoomIn(255) _
           +VectColor(&H801515) _
           +VectFBox(-50,-125,50,125) _
  +VectEndZoom _
  +VectZoomIn(80) _
           +VectColor(1) _
           +VectLine(-50,-125,0,-100) _
           +VectLine(0,-100,50,-125) _
           +VectLine(-50,125,0,100) _
           +VectLine(0,100,50,125) _
           +VectLine(0,-100,0,100) _
  +VectEndZoom _
  +VectZoomIn(10) _
           +VectColor(&H808000) _
           +VectFCircle(25,0,10) _
  +VectEndZoom
```

```
Racer=VectVehicle _
  +VectRem("Simple line when zoomed out") _
  +VectZoomOut(255) _
        +VectColor(1) _
        +VectLine(0,-90,0,80) _
        +VectExit _
  +VectEndZoom _
  +""_
  +VectZoomIn(150) _
        +VectRem("front air dam") _
        +VectColor(&Hc0c0c0) _
        +VectFBox(-30,-85,30,-75) _
  +VectEndZoom _
  +""_
  +VectZoomIn(255) _
        +VectRem("car body") _
        +VectColor(1) _
        +VectFBox(-10,-90,10,70) _
        +VectFBox(-20,0,20,70) _
  +VectEndZoom _
  +""_
  +VectZoomOut(150) _
        +VectRem("Zoom out wheel block") _
        +VectColor(&H202020) _
        +VectFBox(-30,-60,-20,60) _
        +VectFBox(30,-60,20,60) _
        +VectExit _
  +VectEndZoom _
  +""_
  +VectZoomIn(150) _
        +VectRem("wheels") _
        +VectColor(&H000000) _
        +VectFBox(-40,-65,-25,-35) _
        +VectFBox(40,-65,25,-35) _
        +VectFBox(-55,65,-25,35) _
        +VectFBox(55,65,25,35) _
        +VectRem("rear air dam") _
        +VectColor(&Hc0c0c0) _
        +VectFBox(-50,70,50,95) _
  +VectEndZoom _
  +""_
  +VectZoomIn(30) _
        +VectRem("pilot") _
        +VectColor(&H1010a0) _
        +VectFCircle(0,12,10) _
        +VectRem("engine") _
        +VectColor(&H606060) _
        +VectFBox(-8,30,8,65) _
  +VectEndZoom
```

```
var ZoomTest = _
  VectZoomOut(250)+VectColor(&H800000)+VectFBox(-125,-125,0,-100)+VectEndZoom _
  +VectZoomIn (250)+VectColor(&H800000)+VectFBox(0,-125,125,-100)+VectEndZoom _
  +VectZoomOut(150)+VectColor(&H008000)+VectFBox(-125,-100,0,-75)+VectEndZoom _
  +VectZoomIn (150)+VectColor(&H008000)+VectFBox(0,-100,125,-75)+VectEndZoom _
  +VectZoomOut( 90)+VectColor(&H800080)+VectFBox(-125,-75,0,-50)+VectEndZoom _
  +VectZoomIn ( 90)+VectColor(&H800080)+VectFBox(0,-75,125,-50)+VectEndZoom _
  +VectZoomOut( 50)+VectColor(&H808000)+VectFBox(-125,-50,0,-25)+VectEndZoom _
  +VectZoomIn ( 50)+VectColor(&H808000)+VectFBox(0,-50,125,-25)+VectEndZoom _
  +VectZoomOut( 25)+VectColor(&H008080)+VectFBox(-125,-25,0,0)+VectEndZoom _
  +VectZoomIn ( 25)+VectColor(&H008080)+VectFBox(0,-25,125,0)+VectEndZoom _
  +VectZoomOut( 15)+VectColor(&H800000)+VectFBox(-125,0,0,25)+VectEndZoom _
  +VectZoomIn ( 15)+VectColor(&H800000)+VectFBox(0,0,125,25)+VectEndZoom _
  +VectZoomOut(  8)+VectColor(&H008000)+VectFBox(-125,25,0,50)+VectEndZoom _
  +VectZoomIn (  8)+VectColor(&H008000)+VectFBox(0,25,125,50)+VectEndZoom _
  +VectZoomOut(  4)+VectColor(&H800080)+VectFBox(-125,50,0,75)+VectEndZoom _
  +VectZoomIn (  4)+VectColor(&H800080)+VectFBox(0,50,125,75)+VectEndZoom _
  +VectZoomOut(  2)+VectColor(&H808000)+VectFBox(-125,75,0,100)+VectEndZoom _
  +VectZoomIn (  2)+VectColor(&H808000)+VectFBox(0,75,125,100)+VectEndZoom _
  +VectZoomOut(  1)+VectColor(&H008080)+VectFBox(-125,100,0,125)+VectEndZoom _
  +VectZoomIn (  1)+VectColor(&H008080)+VectFBox(0,100,125,125)+VectEndZoom


var ZoomRange = _
   VectColor(&H808080)+VectFBox(-25,-125,25,125)+VectEndZoom _
  +VectColor(&H800000) _
  +VectZoomRange(1,2)+VectFBox(-125,-120,125,-90)+VectEndZoom _
  +VectZoomRange(2,4)+VectFBox(-125,-90,125,-60)+VectEndZoom _
  +VectZoomRange(4,8)+VectFBox(-125,-60,125,-30)+VectEndZoom _
  +VectZoomRange(8,16)+VectFBox(-125,-30,125,0)+VectEndZoom _
  +VectZoomRange(16,32)+VectFBox(-125,0,125,30)+VectEndZoom _
  +VectZoomRange(32,64)+VectFBox(-125,30,125,60)+VectEndZoom _
  +VectZoomRange(64,128)+VectFBox(-125,60,125,90)+VectEndZoom _
  +VectZoomRange(128,255)+VectFBox(-125,90,125,120)+VectEndZoom _
  +VectColor(&H000080) _
  +VectFCircle(0,-105,10) _
  +VectFCircle(0,-75,10) _
  +VectFCircle(0,-45,10) _
  +VectFCircle(0,-15,10) _
  +VectFCircle(0, 15,10) _
  +VectFCircle(0, 45,10) _
  +VectFCircle(0, 75,10) _
  +VectFCircle(0, 105,10)
```