550 388 01

Visual VIGO version 1.06 General Help

Manual

GB

PROCES-DATA A/S

NAVERVEJ 8, DK-8600 SILKEBORG • Tel. +45-87 200 300 • Fax +45-87 200 301 • info@proces-data.com • www.proces-data.com

550 388 01

August 2001

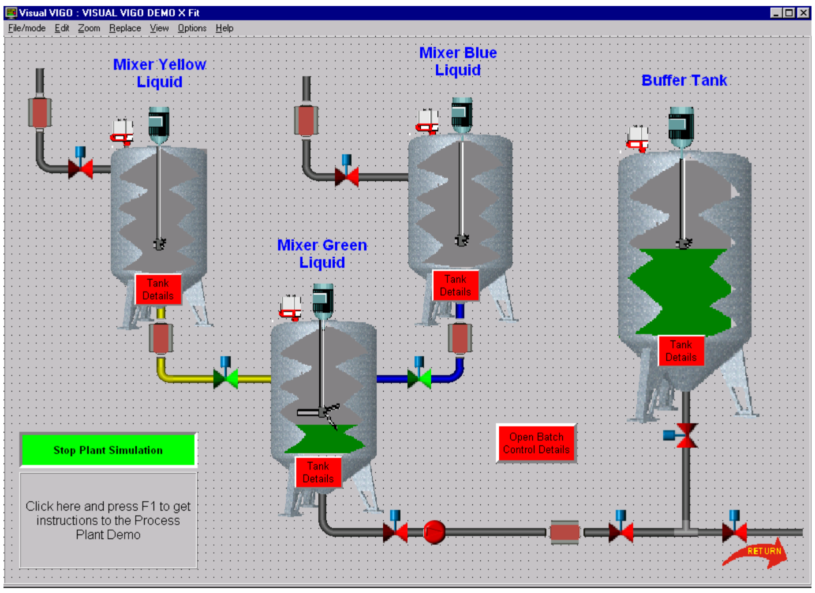
Contents

1 Visual VIGO design time help

1.1 General Features

Visual VIGO is a Windows based SCADA (Supervisory Control and Data Acquisition) application program, used to visualise and operate process plants, factory automation, building management systems, etc. on a PC. The PC can be connected on-line locally, or remotely via a LAN, direct MODEM access or Internet connection, through the VIGO field bus management system and the P-NET fieldbus.

A process plant is visualised by designing one or more *Work floors*, where machines, equipment and instruments can be placed. The operator is able to “walk” through the factory by opening “doors” to the different *Work floors* and departments of the Factory. The different machines and instruments can be opened for studying internal details.


Visual VIGO : VISUAL VIGO DEMO X Fit
File/mode Edit Zoom Replace View Options Help
Mixer Yellow Liquid
Mixer Blue Liquid
Buffer Tank
Mixer Green Liquid
Tank Details
Tank Details
Tank Details
Tank Details
Stop Plant Simulation
Click here and press F1 to get instructions to the Process Plant Demo
Open Batch Control Details
RETURN

Visual VIGO is based on an object-oriented principle, where *Visual VIGO components* can contain other *Visual VIGO components*, each of which can contain other *components* etc. *Visual VIGO components* are placed on the *work floors*, and can be used to illustrate a complete working floor, a machine, a detail of a machine, an actuator, a sensor, a control function, a value, a set point, a data recorder etc.

The way a *Visual VIGO* component is presented to the user involves the use of images, multimedia items and help information. *Visual VIGO components* can take different states. These states can be represented by using a set of chosen images, and if required, include sounds and visual effects. The images could originate from a picture derived from a camera or scanner, be generated by any standard drawing program or computed by *Visual VIGO components*. Multi-media effects are based on standard .avi and .wav type files, either generated by the user or chosen from a variety of prepared sources. If required, Help information can be generated by the designer using standard text preparation facilities and associated with a particular component.

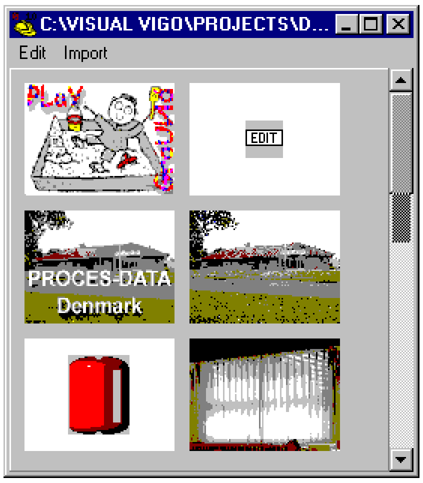
The following criteria have been the main goals for *Visual VIGO:*

- Ease of use for designers as well as operators.

- System integrators can customise *Visual VIGO components*.
- *Visual VIGO components* can be developed by third parties, and added to existing systems without compilation.
- *Visual VIGO* runs within the VIGO environment, and as such, *Visual VIGO components* can inherit the functionality of VIGO.
- The principle of using the right-hand mouse button in VIGO , where a selected channel or variable shows a menu with the relevant tools or properties, is the same in *Visual VIGO*.
- The right mouse menu can also be used to open relevant documentation, such as Help files and hardware manuals, including sensors, actuators and related connection diagrams.

1.2 Ease of Use

One of the main goals of *Visual VIGO* has been that it would be so easy to use, that the need for reading the manual would be minimised. This has been achieved by adopting the same familiar methods for editing elements as used within other standard Windows applications. As an example, the following functions can be found in right-hand mouse button menus: Cut, Copy, Paste, Delete, Group, Order, Edit of properties and Help.



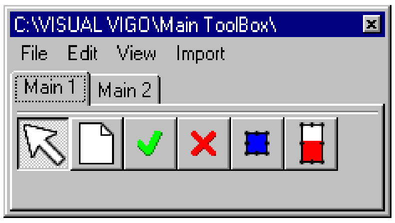
From a right-hand mouse menu, the *Component Tree* can be opened to inspect and change the configuration of *Visual VIGO components*. Such configuration could relate to properties of the *Visual VIGO components* or parameters in a related I/O channel in the hardware.

In *Visual VIGO*, the need to deal with file and folder names has been reduced to as low a level as possible. The designer is only required to define names in two situations:

- A complete *Visual VIGO* design is saved under one file name.
- A *Component box*, together with all its components, is saved under one folder name.

Some of the images used in a *Visual VIGO* design may be based on captured images. All such images contained in a design can be shown within a *Presentation browser*, and be selected from the browser, without using file names.

The images available within *Visual VIGO* are automatically given unique identities, to avoid accidental overwriting of images that may have had the same file name.



A *Visual VIGO* design is created by copying *Visual VIGO components* into the design from a *Component box* or from another *Visual VIGO* design. All components are independent. New components can be copied into the *Component box*, **without** the requirement of compiling any programs: just copy and paste the components. Adding new *Visual VIGO components* in this way, makes it possible to introduce new *Visual VIGO components* without the risk of changing the functionality of existing components.

Visual VIGO components can be easily exchanged between different *Visual VIGO* users. This is done by first copying the components to be exchanged into a *Component box.* From the *Component box*, all components can be packed into one folder, and sent by email.

The *Component box* and its contained components are not required when the design is being used in *Run mode*.

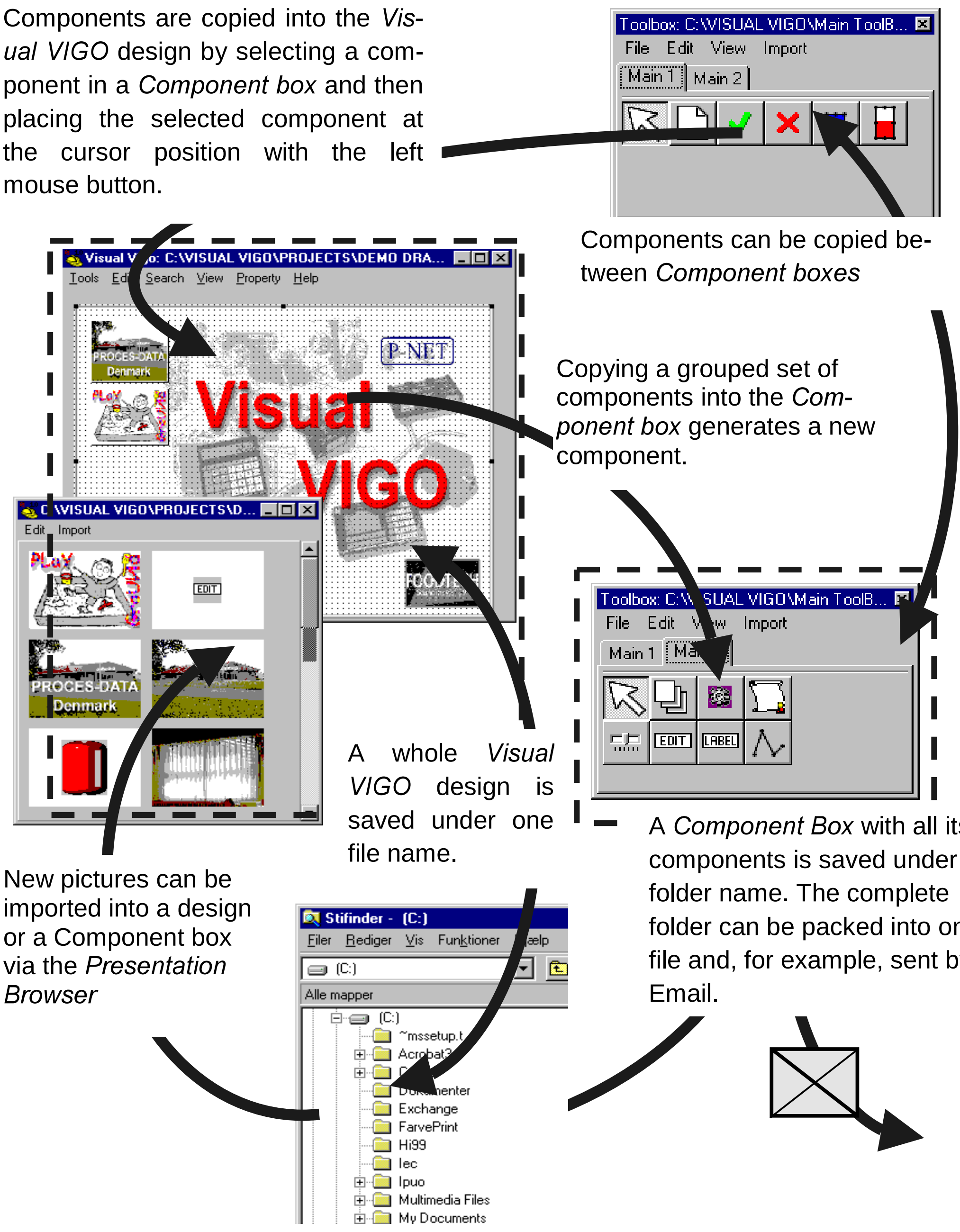
Visual VIGO is designed for utilization at different levels for:

- Operators using a *Visual VIGO* design created by others.
- Designers that create new *Visual VIGO* designs based on standard components.
- Designers that create new components by grouping a set of standard components and then copying them to the *Component box*.
- Component designers that create new components with a new appearance, but having the same functionality. For example, a Valve component can be converted into a Pump component just by exchanging the images.
- Additional components having a new functionality can be developed using Delphi from Borland, using the *Visual VIGO* class library.

Visual VIGO keeps track of which components are visible, and which are hidden behind other *work floors*. In this way, time wasted spent in reading variables over the network to update values on the screen that are not visible, can be avoided. The consequent bandwidth requirement for the network is therefore heavily reduced. This also leads to the fact that invisible components can release memory allocated to store images, and thus reduces the overall memory requirement. The VIGO objects can be released when the component is hidden, and as such can, for example, close Modem connections.

1.3 *Visual VIGO* Edit scenario.

Components are copied into the *Visual VIGO* design by selecting a component in a *Component box* and then placing the selected component at the cursor position with the left mouse button.



Components can be copied between *Component boxes*

Copying a grouped set of components into the *Component box* generates a new component.

A whole *Visual VIGO* design is saved under one file name.

A *Component Box* with all its components is saved under one folder name. The complete folder can be packed into one file and, for example, sent by Email.

New pictures can be imported into a design or a Component box via the *Presentation Browser*

2 Terminology in design mode

Throughout this help system, certain common terms are used, which have a specific meaning in the context of a particular description. The following paragraphs provide a short definition of such words, phrases or functions. Also, where appropriate, certain typographical styles have been adopted to convey the meaning or usage of certain words.

- Italics are used for words that have a specific meaning in *Visual VIGO,* e.g. *Work floor*
- <> brackets are used to convey a keyboard key name or a button shown on the monitor.,e.g <Enter>
- [ ] brackets are used to convey the field name in a displayed window or a menu item, e.g. [Save as..], [Name:]

2.1.1 Design

Design is the term for a Visual VIGO set-up, made to interact with a control system. A *design* is saved in a single packed file, containing all the necessary data for performing the functionality.

2.1.2 Component

Component is a term for an object that can be placed on a *Work floor*. The *Components* are normally visible and used to indicate the current states of the control system. The *components* have properties with which they can be configured. Each base *Component* is created as an individual piece of software compiled separately as a DLL.

2.1.3 Properties

The functionality of a *Component* is based on value settings of some associated variables. These variables are called *Properties*, and those that can be modified by the user can be adjusted using the *Component tree* editing facilities.

2.1.4 Component box

A *Component box* is a store for components, holding a *component* of each available type. From the *component box*, copies of *components* are moved into a *design*. Each *component* in the *component box* is stored with configured *properties* including pictures, help information etc. New *components* can be copied into the *component box*, using copy / paste. The collection of *components* in a particular *component box* is saved under a single file folder. Extended information can be seen in the *component box* section.

2.1.5 Work floor

A *Work floor* is a *component*, which has a basic surface onto which other *components* can be placed. A new *design* always contains one **main** *Work floor*, onto which all other *components* must be placed. Additional *Work floor components* can be added in a chosen hierarchical structure. This means that one floor can be opened from another floor and so on, to unlimited levels.

2.1.6 Image

An *image* defines the visual appearance of a *Component*. A *component* may vary its *image* to reflect its state. An *image* can originate from a picture taken with a camera, be a computer designed bitmap, be generated by programs, or be simply a colour or pattern.

2.1.7 Component Element

A *Component Element* is a collection of configurable *properties* for a *component*. A particular *Component Element* deals with the setting and holding of values for a range of *properties* that apply to a specific aspect of the *component*. For example, the *Frame component element,* which applies to a number of standard *components*, enables a user to define the style, width and colour of a border around a *component*. Some *Component elements* call upon other *component elements* if the user chooses to utilise additional property settings. The hierarchical structure and relationship between *Component elements* can be seen in the *Component tree* window.

2.1.8 Component tree

A *Visual VIGO Design* has a hierarchical structure, where *components* hold other *components*. This structure can be visualised as a tree, starting with the main *Work floor* as the root, followed by branch components holding other *components* etc. The *Component tree* window shows this structure together with the editable properties of a selected *component*. A *component* can be “selected” by clicking the *component* in the *Visual VIGO Design* or the icon of the *component* in the *Component tree*. When it is selected, it is highlighted in the tree, and the *component* in the *Visual VIGO* container is shown with boundary markers. Additional information can be seen in the *Component tree* section.

2.1.9 Presentation selector

The *Presentation Selector* is a window to see or select items currently included in a *Design* or a *Component box*. Such items can include pictures, icons and bitmaps. If certain items are not yet included, they can be imported via the *Presentation selector*. The selected items are used to present component states, activities or help information to the user, in a versatile and meaningful manner. Additional information can be seen in the *Presentation selector* section.

2.1.10 PhysID

A Physical Identifier (*PhysID*) is a reference to a variable in the external network. *PhysID’s* are defined by *VIGO* in order that other applications (such as *Visual VIGO*) can use these references to obtain or send a real time value to a particular variable. The *PhysID* is one of a number of properties that needs to be defined for many components that are designed to show the state or value of a variable. The *PhysID* property window shows the network structure as defined in the *VIGO MIB*, and a variable is selected in exactly the same way as in *VIGO*.

3 The Visual VIGO Container in design mode

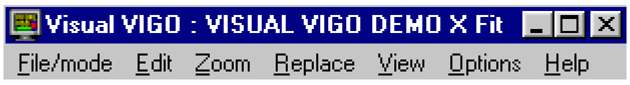
Visual VIGO is a modular system, built up using a *Visual VIGO container* and a number of independent *Visual VIGO components*. The *Visual VIGO container* is physically located in *Visual VIGO.EXE,* some DLLs (Dynamic Link Library) and some images etc. *Visual VIGO components* typically consist of a DLL, together with one or more images, and a help file.

The *Visual VIGO container* can be executed in two modes, *Run mode* and *Design mode*.
In *Run mode,* the main task of the *Visual VIGO container* is to open or close a *Visual VIGO* design and to perform a few service tasks for the components. The most important service task is to activate the components periodically. Most of the functionality of the system is programmed into the included components.

In *Design mode,* the task of the *Visual VIGO container* is to receive edit commands from the keyboard and mouse, to move and copy components, edit properties, undo, redo etc. The components have a number of properties, including size and position. During the edit phase, the components receive new property values, and by this means, their appearance and position can be changed.

Access to any edit facilities depends on the VIGO access settings and whether *Visual VIGO* is in *Run mode* or *Design mode*.

3.1 Main menu description



The figure above shows the main menu, as it appears when *Visual VIGO* is in *Design mode*. In *Run mode*, only the File/mode, Zoom and Help menu items are visible.

3.1.1 File/Mode

The File/mode menu contains the following items:

New
This function will open a new *Visual VIGO* design with an empty container *Work floor*, in a new file with a default name. If another design is already open, this will first be closed. The password for the new design will be empty, and the basic *Work floor* will be sized according to the current screen settings of the PC.

Open
This function is used to open a previous *Visual VIGO* design. The design is selected from a standard file browser, from where the previously named file is selected. If another design is already open, it will first be closed, after determining whether changes have been made.
If the design is changed but not saved the designer will get the opportunity to save it before the new design is opened.

Close

This function is used to close the current *Visual VIGO* design. If the design has been changed since it was last saved, the designer will be asked whether he wishes to save it or not.

Print

This item opens a window giving screen image information about a complete design or a selected *Work floor*. It also provides facilities to size, save as a .jpg image file, to preview, to set up a printer and to activate a print. See How to use Visual VIGO section.

Save

This function is used to save the current *Visual VIGO* design. It saves the design under the file name already given. If a file name has not previously been chosen, the user is asked to select a file name from a browser or to key in a new one.

Save as

This function is used to save the current *Visual VIGO* design under a new file name and thus make a copy. The file name is selected from the browser or a new one is keyed in.

Design

This menu item is used to change between *Run mode* and *Design mode*. The initial state of the menu is that this item is ticked when a new design is created, meaning *Visual VIGO* is in *Design mode*, where editing of the new *Visual VIGO* design can take place. To change the mode, just click once on this menu item. The state of the current mode is saved along with the design, and will be restored when the design is re-opened. Toggling between Design and Run modes can also be achieved from the keyboard, by pressing <Alt> + <F><D>

In *Design mode,* the *Visual VIGO* design can be edited, whilst at same time, all the *Visual VIGO components* are active and perform their normal function, including reading values over the network. The only difference compared with *Run mode*, is that the mouse buttons change their functionality, although the Run mode functionality can be achieved in Design Mode by holding down the <Ctrl> key while activating the mouse buttons.

3.1.2 Edit

The Edit menu contains the following standard functions together with their shortcuts:

Select all <Ctrl>+<A>

This function selects all *Visual VIGO* components on a selected floor or in a selected group.

Cut <Ctrl>+<X>

This function removes the selected *Visual VIGO component(s)* and stores a copy in the Windows clipboard, including all associated properties.

Copy <Ctrl>+<C>
This function copies the selected *Visual VIGO component(s)* to the clipboard, including all associated properties.

Paste <Ctrl>+<V>
This function will place the formerly cut or copied *Visual VIGO component(s)* onto the selected *Work floor*. Clicking the right mouse button when the cursor is positioned on a *Work floor* can also open the edit menu. When this is done, and *Paste* is selected, the component will be placed at the cursor position.

Delete <Del>
This function removes the selected *Visual VIGO component(s)* without copying it to the clipboard, meaning that the contents of the clipboard remain unchanged.

Undo <Ctrl>+<Z>
This function can undo the previously described edit functions (up to 100 actions), as long as changes to the design have not yet been saved. If an automatic save is required for a particular action, the user will be warned that this cannot be undone. The undo function also applies to property editing.

Redo <Ctrl>+<Y>
This function can redo the previously described undo function, as long as changes to the design have not yet been saved.

3.1.3 Zoom

This menu item enables the size of the screen image of a design to be expanded or contracted to suit both design and run mode activities.

In <Ctrl>+<I>
Makes the screen image larger, in order to focus in on a particular area of the design. When the design becomes larger than the screen, scroll bars will appear. A particular area can be centralised by using the scroll bar controls or the cursor control keys.

Out <Ctrl>+<I>
Produces the opposite effect to In.

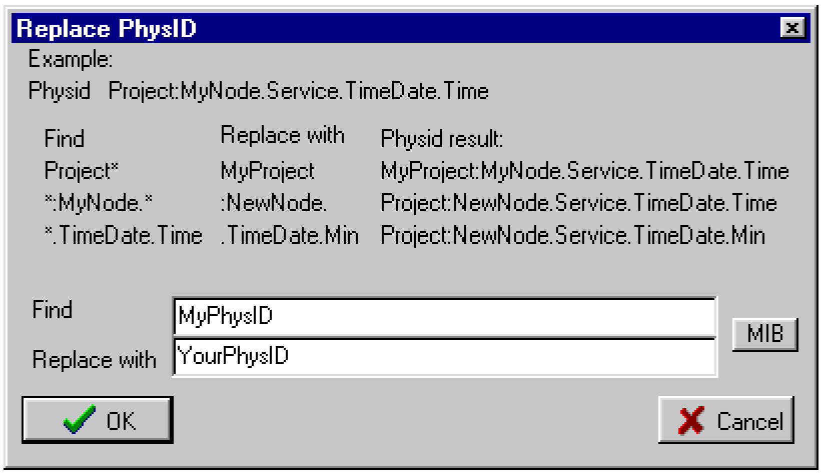
Normal <Ctrl>+<N>
Resets the screen image to the original default size.

Fit to window <Ctrl>+<W>
Resizes the design to fit into the currently displayed area of the window.

3.1.4 Replace

The Replace menu has the following items:

PhysID


Replace PhysID

Example:

Physid Project:MyNode.Service.TimeDate.Time

Find	Replace with	Physid result:
Project*	MyProject	MyProject:MyNode.Service.TimeDate.Time
:MyNode.	:NewNode.	Project:NewNode.Service.TimeDate.Time
*.TimeDate.Time	.TimeDate.Min	Project:NewNode.Service.TimeDate.Min

Find: MyPhysID

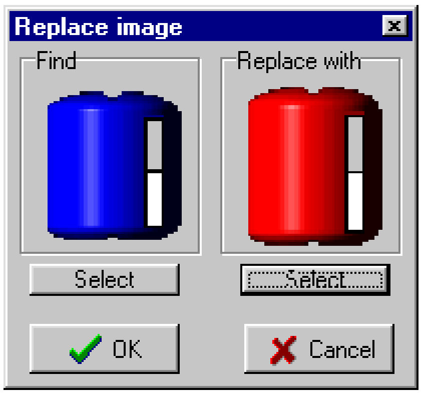
Replace with: YourPhysID

This function can replace the whole or part of the PhysIDs located in a number of selected *Visual VIGO components* at the same time. It is used to change the project name or node name and accepts wildcards of the form “*TEXT*”. This is very useful if a design is moved or copied from one PC to another, where the path to a physical variable may be different, or where certain components have initially been using test or simulation variables, but now need to be directly connected to a plant.

The text in [Find] is searched in the PhysIDs of all selected components, and where a match is found, it will be replaced with the text in [Replace with].

Image

This function can replace the reference to a defined image used by various selected *Visual VIGO components*. This is done by selecting “before” and “after” images from the *Presentation selector*. By clicking the <Select Find> image, the *Presentation selector* is opened and the “before” image is selected. The same method is used to <select> the “after” image. The replace image option will only work if a symbol in the design is selected. If a *Workfloor* or a *Group* symbol is selected the *replace image* option will affect all components included into the selected *Workfloor* or *group*



Dll’s

This function can update the DLL of a *Visual VIGO Component* with a newer version of the same type. DLL names are made up of a unique ID followed by a version number. Only DLLs with an identical ID will be updated. When this function is activated, a normal file browser is opened for selecting the new DLL.

3.1.5 View

The View menu contains the following items:

Presentation selector

This menu item is used to open a *Presentation selector*. The *Presentation selector* shows all the images used in a particular *Visual VIGO* design or *Component Box*, and is used for selecting and importing such items (see Presentation selector).

Component box <F6>

This menu item is used to open the default or last used *Component box*. Additional *Component boxes* can then be opened or created and a new default selected. A *Component box* contains copies of *Visual VIGO components.* Components are copied from the active *Component box* into the design. For a detailed description, see Component Box section.

Component connection points

Some *Visual VIGO components* may support functional *connections* to other components. A *connection* is established by drawing a line between the *connection points*. This menu item is used to select whether, for such components, the *connection points* and lines are visible.

Component tree <F5>

Visual VIGO components are configured by means of a set of properties, which are set individually for each component. The *Component Tree* is used for this purpose. Click once on this item to open the *Component tree* together with its *Property detail window* for the selected component. Whilst this item is checked, the property editor window will remain visible for any other component selections. The Component tree can also be opened from the keyboard by pressing <Alt> + <V><M>. For a detailed description, see Component Tree section.

Errors during start

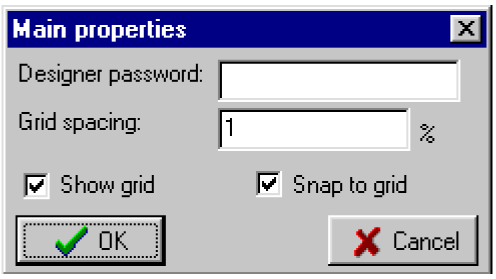
Instead of showing errors individually during load of a design or initialisation of a *component* in the *design,* the errors are collected in an *error list*. When the loading of a *design* or creation of a *component* in the *design* is completed, the *error list* is displayed if there are any errors to report.

3.1.6 Options

The Options menu contains the following items:

Main Properties

This menu item opens a window dealing with aspects about this particular design. It provides the means to set and show a grid on all *Work floors* within the *Visual VIGO* design. The grid can be made to be visible or invisible. It can also be defined as to whether movements of components will “Snap to grid” or not. If the snap facility has been selected, it can be temporarily disabled while the Alt key is held down.


Main properties

Designer password:

Grid spacing: 1 %

[✓] Show grid

[✓] Snap to grid

The window also provides the ability to set a password. After initially creating a new design, the password will be blank. Once a password has been devised and entered here, Design mode cannot be accessed from Run mode without first entering the password.

Optimise design

When a component is copied into a *Visual VIGO* design, all necessary DLLs, help files and images that are not already included in the design, are copied as well. It is also possible to copy images into the design for later use. When a component is deleted, it is not tested as to whether there are DLLs, help files or images that are no longer in use. When the design is finished, it may therefore hold unused DLLs or images. The optimise function will remove all the unused files.

3.1.7 Help

This Help menu contains the following items:

Visual VIGO

This item provides information about general VIGO functionality. Help information about an individually selected *Visual VIGO component* is opened by pressing <F1>.

The following sections are included in the *Visual VIGO* main help system:

- Introduction: Explains the general use of *Visual VIGO.*
- Terminology: Describes the various terms used in the *Visual VIGO* help system.
- Help System: Contains a detailed description of the help system.
- Right mouse menu: Describes the contents of the right mouse menus, both in *design mode* and in *run mode*.
- Component tree: Describes the use of the component tree in *design mode*.
- Component elements: Describes the basic features of component elements.
- Component box: Describes the features and use of component boxes.
- Components: Describes the basic concept of components.
- Presentation selector: Detailed description of the image browser.
- How to use Visual VIGO: Description of how to operate *Visual VIGO*.
- Making and Developing New Components: Describes how to create custom designed components.
- Evaluation and Tutorial: Information about examples and tutorials.

Main run time help

Opens the actual help file linked to the design.

About Visual VIGO

Opens a window not only providing details about the version number of the application, but also shows the number of components that are allowed in a design without an additional licence. It also shows a count of the number of components currently being used.

3.2 Right mouse menu

When the cursor is located over a *Visual VIGO* component including a *Work floor*, and the right mouse button is clicked, a menu is opened. The type of menu depends upon whether the *Visual VIGO container* is in *Design mode* or *Run mode*.

3.2.1 Right mouse menu - Design mode

In *Design mode*, the following menu is opened.

Component tree

Visual VIGO components are configured by using a set of properties, which are set individually for each component. The *Component* tree provides all the editing facilities required for this purpose. By selecting the Component tree from this menu, even if it already open, will highlight the name of the selected component within the tree and display the “Name” component element editing window. For a detailed description about the Component tree and how to use it, see *component tree* section.

Component box

This menu item provides a convenient alternative means of opening the Component box for new component selection. For a detailed description about the Component box and how to use it, see *component box* section.

Cut <Ctrl>+<X>

This function removes the selected *Visual VIGO component(s)* and stores a copy in the clipboard, including all associated properties.

Copy <Ctrl>+<C>

This function copies the selected *Visual VIGO component(s)* to the clipboard, including all associated properties.

Paste <Ctrl>+<V>

This function will place the formerly cut or copied *Visual VIGO component(s)* onto the selected *Work floor* at the cursor position.

Delete <Del>

This function removes the selected *Visual VIGO component(s)* without copying it to the clipboard, meaning that the content of the clipboard remains unchanged.

Zoom

Provides the ability to focus on a particular area of the design screen.

- In <Ctrl>+<I>: Makes the screen image larger in order to focus in on a particular area of the design. When the design becomes larger than the screen, scroll bars will appear. A particular area can be centralised by using the scroll bar controls or the cursor control keys.

- Out <Ctrl>+<O>: Produces the opposite effect to In.

- Normal <Ctrl>+<N>: Resets the screen image to the original default size.

Grouping

Visual VIGO supports grouping of selected *Visual VIGO components*. Grouping of *Visual VIGO components* locks the relative position between them, and the whole group can be moved, copied, cut, and pasted. The functionality of the individual *Visual VIGO components* remains unchanged, and each component in a group reacts individually to mouse activity. A group can be grouped into another group to unlimited levels.

- Group: This function will include the selected *Visual VIGO components* in a group. *The Component tree* will recognise the grouping and provide the facility to name it. Grouping is often used to generate new components.

- Ungroup: This function will dissolve the selected group. The *Visual VIGO components* of the dissolved group can now be individually moved. Groups within the dissolved group will remain grouped.

Order

The graphical *Visual VIGO components* on a *Work floor* are placed in different layers, one component in each layer. Normally, the latest inserted *Visual VIGO component* will be on the top or front layer of all the other *Visual VIGO components*. This is only of importance, when *Visual VIGO components* are to be placed on top of others and where the order of each may then need to be considered. The *Component tree* also reflects this order, in that it shows the layering relationship between each component in terms of displaying the rear most layers at the top of the tree (always the default main floor), running forward down the list to the front layer at the bottom of the tree. The following sub-menus can change this order.

- Bring to front: This function will bring the selected *Visual VIGO component* to the top layer of all others on its *Work floor*.

- Send to back: This function will place the selected *Visual VIGO* component on the lowest layer of all others on its *Work floor*.

- One forward: This function will position the selected *Visual VIGO* component one layer forward.

- One back: This function will position the selected *Visual VIGO* component one layer backwards.

About …
Opens a window providing details about a base component such as generic name, brief description, design company, version etc.

Component Specific Menu Items
Some components have additional right-hand menu items to aid with orientation or component referencing. Such specific functions are shown below a dividing line on the menu.

Rotate
This facility provides the means to amend the outline view of the component.

- Clockwise: Turns the component outline by 90 degrees in a clockwise direction for each click.

- Counter clockwise: Turns the component outline anti-clockwise by 90 degrees for each click.

- Mirror horizontal: Causes the outline to flip in a horizontal fashion.

- Mirror vertical: Causes the outline to flip in a vertical fashion:

Referencing
Provides an easy means to define a reference to a particular component. This applies specifically to *Open* and *Work Floor* components, where a link needs to be established as to which Work Floor is to be opened when an Open control is activated.

- Copy Ref: Appears in a *Work floor* menu. When clicked, identifies this *Work floor* as the reference for an *Open* component.

- Paste Ref: Appears in an *Open* component menu. When clicked, makes the link with the previously referenced Work Floor.

Sub Items
This menu item is reserved to provide dynamic information to the designer and user. For example, it will appear in the menu list if there is a problem with communication between the selected component and the linked variable. Selecting this menu item, if it is present, will show some error text relating to the problem.

3.2.2 Right mouse menu - Run mode:

The availability of a right mouse menu in *Run mode* depends on the properties of individual *Visual VIGO components*. At the moment, no menus are available for the standard *Visual VIGO components* accept the Monitor Line component. However, as with the Design mode facility, a Sub Items menu item will appear if there is some kind of communication problem between the selected component and the linked variable. Such a problem would normally be initially indicated by display of the error image for that component. By right clicking on the component and selecting Sub Items, will provide information as to the cause of the error image.

4 The Visual VIGO Help System

Visual VIGO offers a powerful *Design mode* and *Run mode* help system to provide context help to both the designer and the end user.

Design mode help

During *design time*, *Visual VIGO* provides help facilities in three forms, namely context help on each included component and its constituent elements, design container menu items help, and full manual help.

If a particular menu item or sub item is selected and <F1> is pressed, a help viewer will open, providing help on that specific item.

If a particular component is selected in the design window and <F1> pressed, a Help viewer will open showing generic help on that component type. Embedded hyperlinks will guide the designer to additional help on the constituent elements of that specific component.

If the *Component tree* is open and a particular component name is highlighted and the <F1> key pressed while the mouse pointer is within the tree section, the *Help viewer* will open and display the same generic help as described when selecting components in the design window.

If the tree structure is expanded, and one of the *component elements* is highlighted, and the <F1> key is pressed while the mouse pointer is within the tree section, the *Help viewer* will open and automatically scroll to the position in the help document where a hyperlink to details about that particular element can be found.

If the mouse pointer is positioned within the *Component element* property section and the <F1> key is pressed, the *Help viewer* will open and display help information about the particular property window currently showing.

If the mouse pointer is placed within the focus bar of the *Component tree* and the <F1> key is pressed, the *Help viewer* will open and display general help on the complete *Visual VIGO* application. The same facility is available by selecting Help from the design container menu item.

Embedding Help in New Components

Designers of new functional components can also include embedded help within them. Further information about how to design new components and how to embed generic help within them can be obtained from PROCES-DATA.

Run mode help

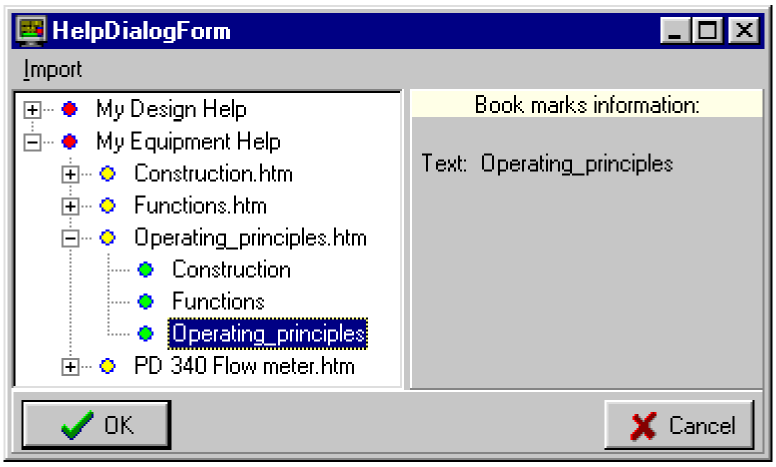
In *run mode*, generic help on component configuration would not be required. In this mode, usage help is provided instead for each component included in the design. However, the highly useful facility is provided for the designer to embed customised help within certain key objects within the operational design. Since no *Component tree* or component configuration is required in this mode, the mouse is the means for determining which help file is displayed within the *Help viewer*. The mouse is positioned over a particular component object and <F1> is pressed. This opens the help viewer and displays a help page related to that object.

Designing Run mode Help files

Basically, *Visual VIGO* help files are generated using a standard word processor, which are then saved in an html (hypertext mark up language) format. This is the format used to publish web pages on the Internet. The designer can chose a specific html document as the main or only help page for a particular component within the design. But this page can also contain hyperlinks to internal bookmarks or other documents. Once the help file has been associated with a particular component, the component can be copied to make multiple instances within the design, and/or pasted to the *component box* for use in other designs or for transmission to other designers. Thereafter, the use of this component will include the facility to open the published help page.

The generation of the files from a word processor is really quite straightforward, although depending on the package being used, there may be different methods and terminology used for including “bookmarks”, and “hyperlinks”. A bookmark is a piece of text within the document (usually a paragraph heading), which is specifically labelled with a bookmark name that can be specified by a hyperlink. This label could have the same name as the heading or some other name chosen by the author. A hyperlink is also a piece of labelled text within the document, normally shown underlined and in a prominent colour. This could also be applied to a heading, but could just as usefully be a single word or phrase within a sentence, which offers the reader the opportunity to jump, by means of a mouse click, to another part of the document, another local document, or an external document held somewhere on the Internet. Therefore, a hyperlink requires that an address is allotted to it, which could consist of the name of a local bookmark, or a path, file name and, if required, bookmark name of a local file, or the URL (Uniform Relation Locator) of a remotely located web page. The designer is advised to consult the manual of his chosen word processor, to become familiar with creating such documents, incorporating bookmarks and hyperlinks, and saving them in html format. It should be remembered that if a document contains pictures and/or diagrams, then when saved as an html page, the word processor will generate an html file plus a folder of a similar name containing all the images. It is important to always keep the file and folder together, otherwise the images will not be displayed.

Once the page/s have been generated they should be saved within a single package of associated html documents under one folder/directory name depicting the type of help package it is. With the *Component tree* open and with any component name highlighted, click on the <Run time help Select> button in the property window. This will open the *help file selector* that displays the imported packages. The tree view holds 3 levels of information, the first level displays the packages, the second level displays the html files located within the packages and finally the last level displays the bookmarks defined in the html files. On the right hand side of the help file selector window, information belonging to the selected object in the tree view is displayed.

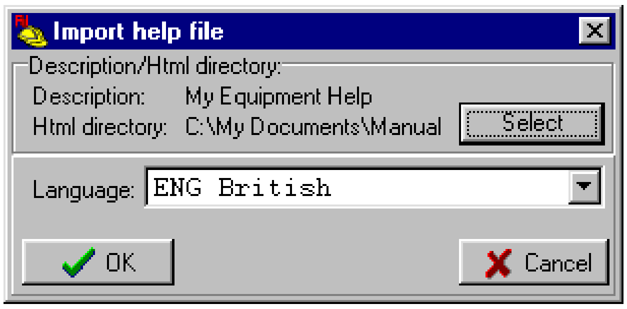


My Design Help
My Equipment Help
Construction.htm
Functions.htm
Operating_principles.htm
Construction
Functions
Operating_principles
PD 340 Flow meter.htm

Book marks information:

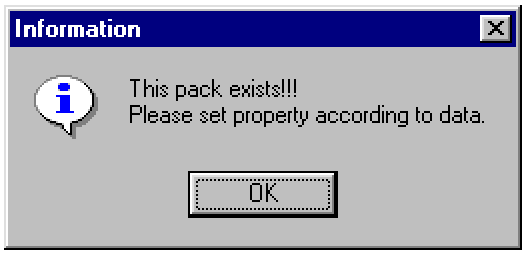
Text: Operating_principles

If this is the first help package to be associated with a component in this design, there will not be any packages displayed. If other help packages are installed, they will be shown in the tree view. By clicking on the menu item <Import> the designer is presented with the opportunity to import new packages.



Description/Html directory:
Description: My Equipment Help
Html directory: C:\My Documents\Manual
Language: ENG British

This window will be shown, which is the means to import your package of html pages, held in a specific folder/directory. First select the language you wish this package to be associated with by using the selector. N.B. You can have different packages associated with different Windows language settings on the operational PC. For example, for a given design, you could have one help package written in English and one written in Danish. Depending on the language setting of the operational PC, the help page in the appropriate language will be used when the user requires help. Press the Select button to open a standard file selector. Select the Folder/Directory containing the html pages and press OK. The name and path to the package Directory/Folder will be shown. Pressing OK will revert to the *Select help file* window, and include this new package in the list. If the designer wishes to utilise the multi-language facility, another package of html documents will be required to be produced using the same html file names, bookmarks and hyperlinks, but obviously written in an alternative language. In this case, rather than selecting a blank line in the *Help file selector*, select the package that has already been installed for the original language, and select <Update> from the right-hand pull-down menu. When the Import help file window opens, select the file browser and chose the Directory/Folder containing the html pages in the other language. An information dialogue will be displayed.



This pack exists!!!
Please set property according to data.

This means that *Visual VIGO* has recognised that a help package with this identity and language has already been installed. Press <OK>, and then make sure that the language of the new package is set to the correct one from the <language selector>. Press <OK>. The new package will now be seen in the list but with the new language label.

Right hand mouse menu

Clicking with the right mouse button on an item in the tree view will depending on the selected item present the designer with the following options:

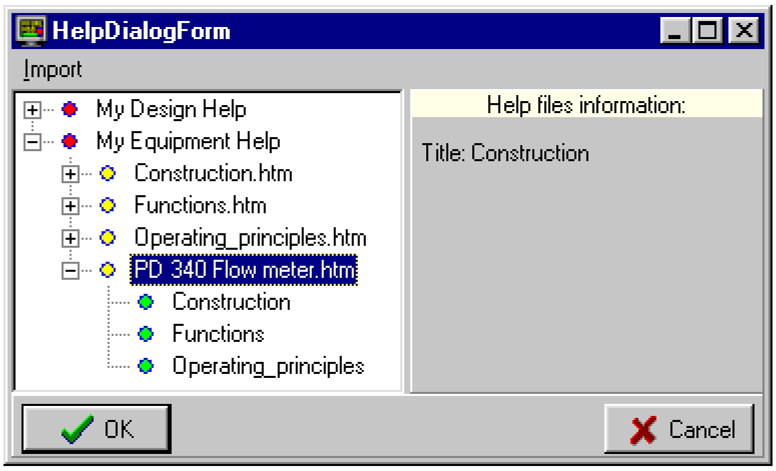
- A HTML file or a bookmark is selected: In this case, the only option will be <Show>. Activating <Show> will launch the HTML file in the H*elp viewer*, and if necessary jump to the selected bookmark.

- A package is selected: In this case, <Update> and <Export> are enabled. Using <Update> will update the contents of a package with new material selected by the designer. <Export> copies all HTML files from the package to a library selected by the designer.

Associating a Component with a help file

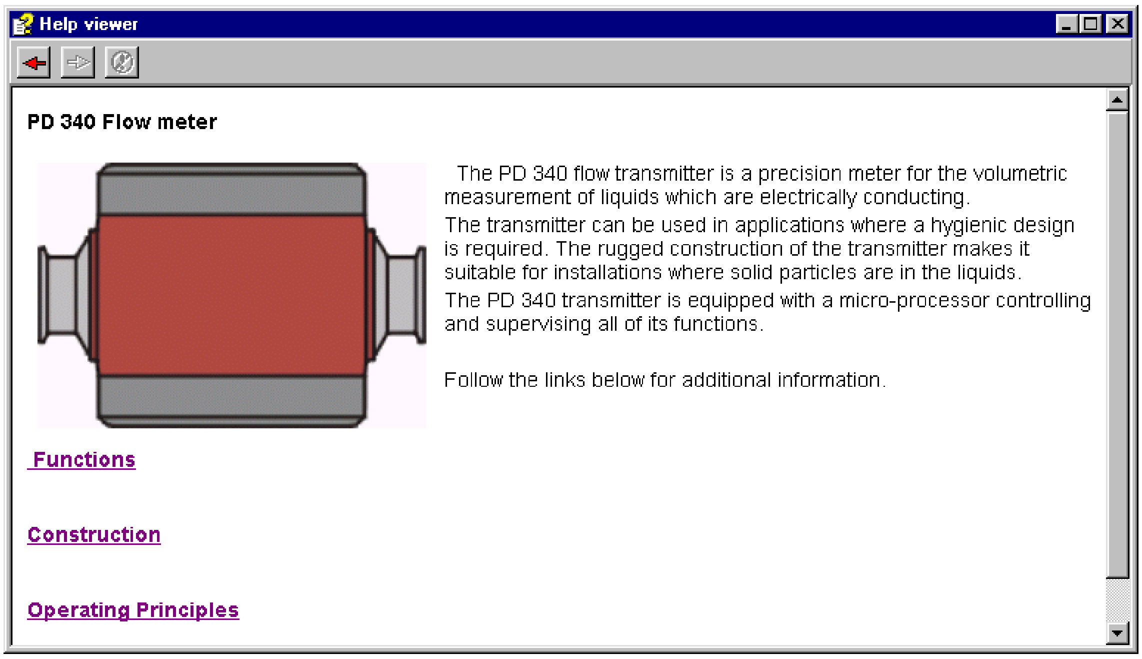
Let us assume that we want to provide some help about a flow meter type that is being used within a design. Let us also assume that some help files have already been written about the flow meter and other equipment, and saved as html pages within Directory/Folders. These packages have already been imported into the Help file selector. A component has been configured to represent the flow meter, perhaps to display a flow measurement or control a flow function.

Open the *Component tree* and select the flow meter component name. Press the <Run time help Select> button, to open the *Help file selector*. Expand the “My Equipment Help package”. This will show all the html files included within this package.



Select the html page you would like the user to see when he positions the mouse over the component and presses <F1>. In this example, the main help page is “PD 340…” The other pages listed are either linked to the main flow meter page or are for another piece of equipment. Then press <OK>. Go into *run mode* and place the mouse pointer over the flow meter component and press <F1>. The help viewer will open and display the prepared page. Clicking on the links will open other pages. By changing the language of the PC (<Settings | Control Panel | Regional Settings>) to the language of the other help package, and again pressing <F1> when the mouse pointer is over the flow meter component, will display help in that language instead if a help package in the selected languages is imported. In *Design mode*, this component can now be copied and pasted within the design or to the component box, but each will retain the help information selected. Performing this operation will override the use of the default run time help file, although the designer can still choose to include this file by including a hyperlink to “Details” in the new run time help file.

Visual VIGO Help viewer



PD 340 Flow meter

The PD 340 flow transmitter is a precision meter for the volumetric measurement of liquids which are electrically conducting.
The transmitter can be used in applications where a hygienic design is required. The rugged construction of the transmitter makes it suitable for installations where solid particles are in the liquids.
The PD 340 transmitter is equipped with a micro-processor controlling and supervising all of its functions.

Follow the links below for additional information.

Functions

Construction

Operating Principles

The *Help viewer* is used in both *Design* and *Run modes*. In *Design mode* it will display help information appertaining to the system in general and for component types and their elements. In *Run mode* it will display help pages associated with the usage of general components and customised help for a selection of specifically chosen operational components. The *Help viewer* displays the contents of the html documents imported into the help file selector within the design. Since it deals solely with html files consisting of text, images, hyperlinks and URL’s, it can be regarded as having a number of the same features as found in a familiar Internet Browser such as Internet Explorer or Netscape. Indeed, if an Internet World Wide Web URL is embedded into a help page and is clicked upon, the help viewer will display the defined web page, assuming dial up networking and a modem is connected to the PC.

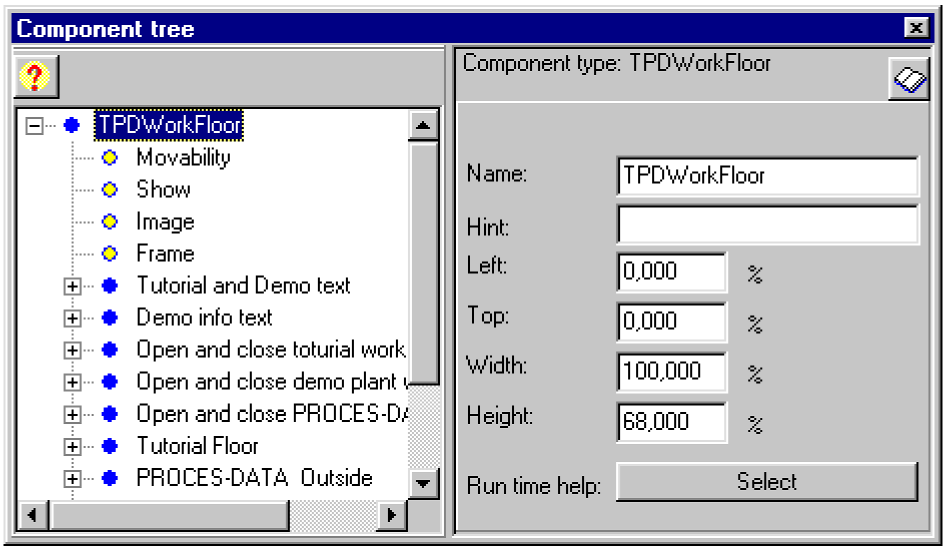
The viewer can be resized, minimised and maximised. It also has a control for stepping forwards and backwards over previously displayed pages. Other features can be found by right clicking on the display where a menu will appear to perform such functions as printing, copying, saving pages as shortcuts etc.

5 Component tree

This section describes the facilities provided to view the structure of a design, and to change how components operate and are presented.

5.1 General Concept

A *Visual VIGO Design* contains a number of *Components*, e.g. work floors, bar graphs, track bars, text boxes, state controls etc. Each component is constructed from a number of *Component elements* Each element contains a single or a number of associated configurable properties, the values of which give the component its own unique characteristic. For example, a standard two state control component needs values to be set for how it will be presented to the user when it is in an “off” state, and how this will change when it is “on”. It may also need to be linked to a physical variable somewhere on a network (PhysID). Due to the versatile nature of Visual VIGO, a particular component can offer all sorts of property settings, including frame types, background colours and images, a wide variety of fonts and sizes for text, and even include multimedia, such as sound and video.


Component tree

- TPDWorkFloor
  - Movability
  - Show
  - Image
  - Frame
- Tutorial and Demo text
- Demo info text
- Open and close toturial work
- Open and close demo plant
- Open and close PROCES-DA
- Tutorial Floor
- PROCES-DATA Outside

Component type: TPDWorkFloor

Name:	TPDWorkFloor	
Hint:		
Left:	0,000	%
Top:	0,000	%
Width:	100,000	%
Height:	68,000	%
Run time help:	Select	

As a design evolves, the number of included components will increase. If a number of *Work floors* are involved, each containing their own associated components, or if a collection of components have been grouped together, it could become a little difficult to remember the relationships between included component types and/or to make quick changes to property values as the design progresses. To this end, the *Component Tree* has been devised. This independent, dual-purpose window is opened from the main menu under View, or via the right-hand mouse menu for a selected component.

When opened initially, the component that is currently selected in the design will be highlighted in the tree with its chosen identity. The position or order of the name in the tree will indicate its position within the layering structure of the design, with the main (default) work floor at the top, being the first component on the initial layer. Each included component is shown with a blue bullet icon and a “+” expansion marker. As additional components are copied to the design from the component box or from within the design, they are added to the tree in their order of generation. Initially, each new component is given a unique name, derived from the type identity of the particular component e.g. Floor 1, Bar Graph 3 etc. However, this name can be easily changed by the designer to reflect its usage. Since a *Work floor* is a container for other components, it can be expanded to show what components it contains. As additional *Work floors* are added, they also can be expanded to show their contained components, or contracted to reduce the complexity of the tree. The method used is very similar to that used in a file manager, where folders are opened to see contained files or other folders, or indeed that used to show the physical nodes and channels included in a *VIGO* MIB. As each level is opened, other contained components are seen, each represented by a blue bullet, some of which may contain more components, and so on. However, unlike a file manager, the *Component tree* provides a highly useful facility to either select a particular component in the tree and see this component selected in the drawing, OR to select a component in the design and see it highlighted in the tree. This feature is invaluable for quickly establishing the identity and structural position of any particular component. Furthermore, when selecting a component in the drawing, the tree will automatically open to the appropriate level in order to highlight it.

5.2 Component Element configuration

So far, we have only described the tree in terms of illustrating its “root” and “branches”. However, it will have been seen that there is an associated Component element detail window attached to the *Component tree*. This initially shows the [Name] and [Hint] Properties of each selected component. The values of either of these properties can be changed in this window. For instance, it may be helpful to change the default name of a bar graph component from say TBarGraph_6 to a more recognisable tag name of “Tank 15 Level”. Also the Hint value, i.e. a label that will appear when the mouse cursor is place over the component, may usefully changed to “Oil Level in Tank 15”. Thereafter, the identity (tag name) that appears within the tree will immediately reflect the change, as will the hint text that appears in both Run and Design Modes.

Obviously, there are many more component elements associated with each component than just its name and optional hint. To obtain a fuller picture of the elements which make up each component, but also their associated property possibilities, click on the <+> expand icon for a particular component. A number of child elements having component element identifiers will be shown for that component, each marked with a yellow bullet icon. By selecting one of these property elements, will display various options in the *Property detail window*. These may require text, selection of a style, deciding on an image type etc. Some *Component elements* may offer deeper choices indicated by further expand icons. The tree detail can be contracted either by clicking individual reduction icons (<->), or by closing and opening the tree again. It will be seen that many of the Component element property windows shown during this process are structured in the same form for many different components. Because of this commonality, each are discussed individually in the Component Element section.

Although each selection of a component and its individual elements can be made by clicking and scrolling with the left hand mouse button, the designer may find it a lot quicker and easier to use the cursor control (<arrow>) keys. The Up and Down keys will highlight each component and displayed element in turn, and the property editor window will change its content to suit each key action. The right and left cursor keys will respectively cause any unopened levels to be opened or any opened levels to be closed.

5.3 Grouped Components

There are many situations where two or more components need to be grouped together, either just to perform a move, or to form a grouped component for use in the current design and/or to copy into the component box as a new component.
To select a number of components at the same time, to form a grouped component, select the first one with a left click and then hold down the shift button while clicking on additional components. The extra components will display gray boundary markers, indicating that they can no longer be resized, but the joint selections can then be moved together. An alternative way of selecting multiple components which are in close proximity, is to hold the Shift key down while holding down the left-hand mouse button, and moving the mouse such as to form a rectangular box around the required selection of components. This will produce gray boundary markers around the composite selection, indicating that the group can be moved together but not resized. While this new group is still selected, additional individual or groups of components can added to the original multiple selections. All these components will have individual gray boundary markers. To deselect a multiple selection, select another individual component or press the <ESC> key. Use the right-hand mouse menu to *Group* the selection.

If the Component tree is displayed following the grouping process, the tree will now show an additional level, which applies to the group as a whole. This provides the ability for the user to name and to give a hint to the group. Expanding the grouped component will reveal the included individual components.

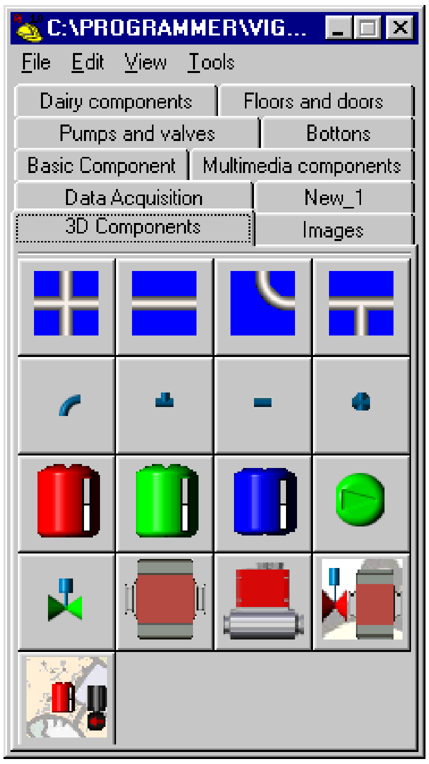
One of the important concepts of *Visual VIGO*, is that each component used in a design can be regarded as a completely independent entity. As such, when a new component is chosen from the component box, an additional set of unique files is generated, which can then be modified individually by the user via its Component element windows. Such a component consists of child elements, the form of which can be the same for a variety of component types, e.g. a component element that defines a *frame* style. In other words, a component is made up of specific elements defining its basic look and functionality, such as whether it is a two state control or a bar graph, together with a number of configurable child elements (group of properties), which enable the user to choose specific functionality, colour, text etc. The consequence of this concept is that it provides the user with enormous flexibility, in that a new component can be derived from a standard type by adjusting size, scale, colour, frame type etc., and then saving it as a new component in the component box for later use in this or other designs. Furthermore, components can be grouped together to form another object. For example, perhaps a new panel instrument consisting of a bar graph, a track bar, an indicator and a push button, could be derived from standard components, modified to suit, grouped together and saved in the component box. Thereafter, this new combination component can then be used over and over again. All the individual *Component elements* will still be visible to the user, in order, for example, to link such an instrument with physical variables within the system. In addition, new components can be designed by the user using the Delphi (Pascal) language. In this case, coding will apply to a particular functionality and visual effect, but common child objects, such as those defining frame style or colour can utilise those used in other components. Alternatively, additional component types can be purchased from third parties, including PROCES-DATA. See the section about making and developing new components.

6 Component element

When configuring components using the *Component tree Property detail window*, the designer will notice that certain named *Component elements* for a variety of component types are presented in the same form and perform a similar job. This is because such elements have been derived from one or a number of generic *Child Objects*, which have been called upon by a particular component. In object oriented programming terms, these *Child Objects* are fundamental in the construction of *Visual VIGO* components. From the point of view of a *Visual VIGO* designer, such *elements* provide the visual means of configuring a single or a set of property values for a particular part of a component. Descriptions, and where applicable, an example of usage within a common *Component element* type for a standard component are provided within the *Visual VIGO* help system. Any of these descriptions can be seen by either selecting the element name in the *Component tree* and pressing <F1>, or by placing the cursor over the element's property window and pressing <F1>.

7 Component box

7.1 General



A *Component box* contains copies of *Visual VIGO components,* including all the necessary associated DLLs, help files and images. It is stored as a single named folder. The *Component box* can be divided into tabs, each holding a number of components. In many ways, a Component box can be regarded as a special form of a design, which is used as a depository for both standard and customised components, except that since the components are not active and there is no relationship between them, no Property tree is involved. Additional named *Component Boxes* can be generated. Such a box may be useful for holding specialised components, or for sending a collection of components to other designers and locations.

When creating a new *Visual VIGO* design, components are picked up from the *Component box* with a single left mouse click, and placed on a *Work floor* at the cursor position when the left-hand mouse button is clicked again. When the component has been placed, it is no longer “held” by the cursor.
A double click on a *Visual VIGO component* in the *Component box* will make the component “stick to the cursor”. This is used for placing several copies of the component on the *Work floor.* The component is removed from the cursor by clicking the right-hand mouse button or pressing the <esc> key.
When <snap to grid> is enabled, the upper left-hand corner of the component snaps to the grid, when it is placed. If the <Alt> key is pressed while placing a component, the snap function is disabled.

The *Component box* can be minimised.

New components can be inserted into the *Component box*, by copying a component from a *Work floor* and pasting it into a *Component Box*. When such a component is later picked up and placed into the same or another design, the property values that were set when originally copied are retained. It is therefore important to ensure that the property values of what can now be regarded as an additional general component are set appropriately before it is copied from a design to a component box. For example, the Variable Link property should be empty, suitable illustrative colours chosen and the component sized so that it is not too large or small.

7.2 Component box menu description

In a similar way to a Design window, a *Component box* window has its own operational menus.

7.2.1 Component box <File> menu

The <File> menu contains the following items:

<New>

This function will create and open a new, empty *Component box*. It is possible to have more than one *Component box* open at the same time, so that components can be copied between them, or that component selection for a design can be achieved from different sources.

<Open>

This function is used to open an existing *Component box*, selected from a file browser. A *Component box* is saved under a folder name. It is possible to have more than one *Component box* open at the same time.

<Close>

This function is used to close the *Component box*. If the *Component box* has been changed, the user is asked whether it is required to save it or not.

<Save as>

This function is used to save the *Component box* under a new folder name and thus make a copy. The new folder name is selected from a browser or a new one is keyed in.

7.2.2 Component box <Edit> menu

The <Edit> menu contains the following items:

<New tab>

This function adds a new empty tab to the *Component box*. Components can later be pasted into the tab page.

<Paste tab> (Open tab)

When the clipboard holds a complete tab, this function will add the tab and its contents from the clipboard to the *Component box*. The properties of the Tab and contained components are also copied.

<Undo> <Ctrl>+<Z>

This function can undo the former described edit functions (up 100 actions), as long as no saving procedure has been performed. The undo function also relates to previous editing of properties.

<Redo> <Ctrl>+<Y>

This function can redo the former described undo functions, as long as no saving procedure has been undertaken.

7.2.3 Component box <View> menu

The <View> menu contains the following items:

<Presentation Selector>
Clicking on this item opens a *Presentation Selector*. This window acts both as viewer and a selection browser for the images associated with components contained within this particular *Component box* folder. It is used to determine how a component is presented to the user when used in a design. The contents of this window would be different if opened from within a design or another *Component box*. Refer to *Presentation Selector* for more detailed information.

7.2.4 Component Box <Tools> menu

The <Tools> menu contains the following items:

<Optimize>
As with a design, any new component added to a *Component box* will be held in the Component box folder. If any components are deleted or removed, the contents of the folder will not be reduced until the Optimize function is clicked.

7.3 Tabs

A Component box can be divided into a number of tabbed pages. Each page can hold a number of *Visual VIGO components*, represented as buttons. The idea of dividing the *Component box* into tabbed pages, is to make it easier to find a particular component amongst many. Each page has its own tab, which can be appropriately named by the user.

7.3.1 Tab right-hand mouse menu

Right clicking on a tab or anywhere on the page where no component is placed, opens a pop-up menu. The menu contains the following items:

<Tab Cut> <Ctrl>+<X>
This function removes the page and its tab from the *Component box* and transfers a copy together with all its contained components to the clipboard. All the properties of the tab and components are also copied.

<Tab Copy> <Ctrl>+<C>
This function transfers a copy of the tab and all its components to the clipboard.

<Tab Paste>
The paste function adds all the copied components to the selected tab page. The function depends on the contents of the clipboard.

- If the clipboard contains one component or a grouped set of components from a *Visual VIGO Work floor*, a new component is inserted into the *Component box*. The image of the component or the group, is copied from the *Work floor* and shrunk to a size that fits as an icon on the button.

- If the clipboard contains a component copied from a *Component box*, the copied component is inserted into the *Component box* with the same icon as that from where it was copied.

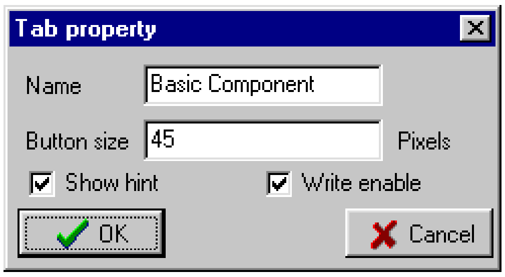
- If the clipboard contains a whole tab page copied from a *Component box*, all the components from the copied tab page are inserted in the selected tab page.

<Tab Del> **<Del>**
This function removes the tab and its contained components from the *Component box*.

<Tab Property>

- The Tab [Name] can be set or changed. The chosen Tab name appears over the tab itself.

- The size (in pixels) of the component buttons within this tab page can be set in the [Button size] field.


Tab property
Name: Basic Component
Button size: 45 Pixels
[✓] Show hint
[✓] Write enable

- If the <Show hint> check box is ticked, the hint associated with each button will be shown when the mouse pointer is passed over them in turn.

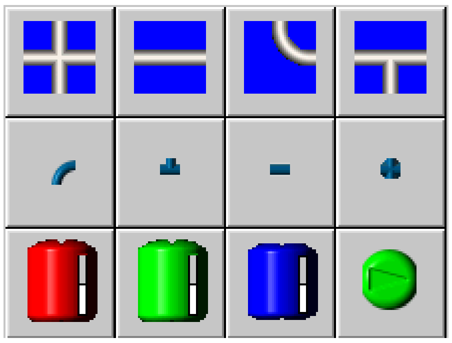
- The contents of the *Component box* can only be changed, if the <Write enable> for the *Component box* is true.

- When the <OK> button is clicked, the changed properties will be used thereafter.

- <Cancel> will retain the properties as they were before opening the property window.

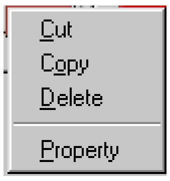
7.4 Component selector buttons

In the *Component box*, buttons with images represent the various included components. The images identify the components visually. A hint can be given when the cursor is positioned above a button. This hint is not transferred to the *Visual VIGO* design



7.4.1 The *Component selector button* right mouse menu

A right click on a *Component selector button* will display a menu with the following items:



Cut <Ctrl>+<X>

This function removes the *Component selector button* from the tab and transfers a copy to the clipboard.

Copy <Ctrl>+<C>

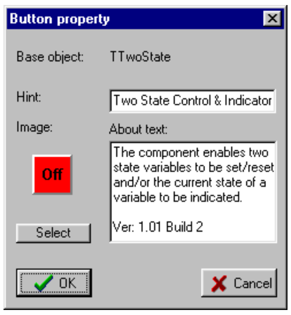
This function transfers a copy of the *Component selector button* to the clipboard.

Delete <Del>

This function removes the *Component selector button* from the tab.
Buttons can be removed or properties changed only if the write enable for the tab is true.

Property

Component selector buttons have a few properties that can be edited in the Button Property window. Properties can only be changed if the write enable for the tab is true.



Base object: TTwoState
Hint: Two State Control & Indicator
Image:
About text:
The component enables two state variables to be set/reset and/or the current state of a variable to be indicated.
Ver: 1.01 Build 2

- A [Hint] can be added to the component selector button, so that helpful descriptive words will appear when the mouse pointer is positioned above the button.
- An [About Text] field is provided for the purposes of reading or inserting more detailed information about the associated component, such as its functionality and version number.
- A thumbnail [Image] of that used with the component is shown.
- A new image can be chosen from the *Presentation Selector* for this Component box but pressing the <Select> button. If the *Presentation Selector* does not contain the required image, the Selector has the facility to import a new one.

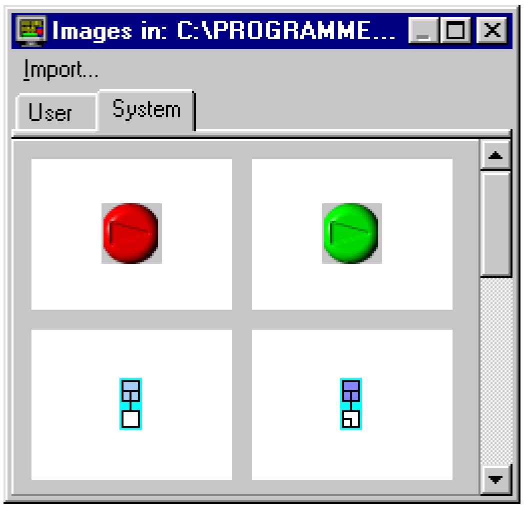
8 Standard Visual VIGO components.

The functionally of *Visual VIGO components* is implemented within individual DLLs. The components have a set of properties, the values of which can be changed and customised to make new components. This means that the same component type (DLL) can, for example, be used for illustrating and controlling a valve, a pump and a push-button, merely by changing some of the properties. The properties are displayed and modified by means of the *Component element* property window within the *Component tree,* opened from the main design menu or the right-hand mouse menu of a component in a design.

A number of standard components are delivered with *Visual VIGO* in the default *Component Box.* Collectively*,* they provide a wealth of functionality to produce simple or complex visualisation and operator intervention designs. Many new components can be produced from these standard objects. Descriptions are included in the *Design time* and *Run time* help systems for each component. The *Component elements* associated with each component are shown, but due to frequent functional commonality, are described in greater detail in the *Component element* section. The descriptions can be seen by either placing the cursor over the component and pressing <F1> (Run Mode), or selecting the component and pressing <F1> (Design mode), or selecting the component in the *Component tree* and pressing <F1> (Design mode).

When used for the first time, each component is given a default name based on the base object name, e.g. TFloor1, 2,3 etc. The designer can rename the component with anything appropriate to its current use, which will then be shown in the Component tree. A “hint” can also be incorporated within any component. This will be displayed when the user positions the mouse pointer over that component. All standard components can be resized using the mouse.

9 Presentation selector



The *Presentation selector* is used to select images for a selected component, in order to present it in a meaningful way to the user. The selector does not actually contain any images but merely acts as a visualisation tool for showing what is already contained within a Design or a Component box, even if it is not currently being used.

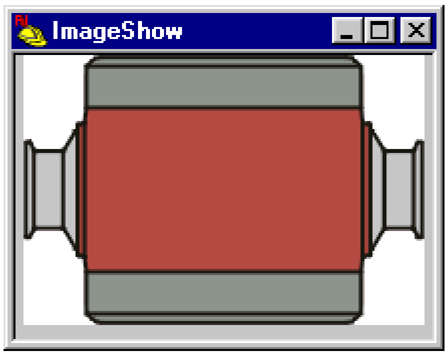
When the *Presentation selector* is opened from the *Design* menu, it shows the images contained within that *Visual VIGO* design. When the *Presentation selector* is opened from a *Component box,* it shows the images contained within the *Component box*.
When the *Presentation selector* is opened from the *Property element editor*, a double click on an image will select the image and close the browser. If an image is subsequently adjusted with the design, such as rotating or mirroring, each stage will be shown in the *Presentation selector.*

The **<Import…>** menu item is used to load new images. From the Import menu, a File browser is opened to select an image of one of the following types: jpg, jpeg, bmp, ico, emf and wmf.

The images can be divided into tabbed pages, and in this way help to provide an improved overview. There is a fixed tab named *System.* All images copied automatically when inserting a new component will be inserted in the *System* tab.
When a new image is imported into the design, it is located at the top of the selector window. The numbers of images shown within the Presentation selector can be reduced by the use of the *Optimise* function, which removes all unused images, help files and DLLs. The original file name of an image is shown as a “hint” for that image in the *Presentation selector.*

The right-hand mouse menu for each Image offers the following functions:

<Show…>



The images shown in the browser may be in a shrunken form, which can make it difficult to see any details. The <show…> function will enlarge the image to the original size.

<Save as…>

The images in *Visual VIGO* are stored as objects with additional information than that held in the original file. This function will save the image in the original format with a specified file name.

<Copy>

This function moves a copy of the selected image to the clipboard. The image can only be pasted from the clipboard into another *Presentation selector*.

<Send to User/System>

Depending on whether the component is in the User or System tab, performs a move operation to the other.

<Delete>

This function is used to delete an unused *Visual VIGO image* or a previously imported image of standard file format. If an attempt is made to remove an image that is currently being used, a warning will be issued and no action taken. Notice that this operation can only be performed when the *Presentation selector* is opened from the main menu.

The menu when right-hand clicking a blank space will show the following:

<Paste>

This function is used to paste the formerly copied *Visual VIGO* image or an image of standard image file format, into the next blank space.

10 How to use *Visual VIGO*

Visual VIGO has been designed to be easy and intuitive to use. However, due to its great versatility and power, there are a few operations and concepts that may not be familiar to new users. This section is designed to describe how to utilize these to the greatest effect.

An important aspect to keep in mind when using *Visual VIGO*, is that is closely bound to *VIGO* – The Fieldbus Management System. It is *VIGO* that provides all the mechanisms to define and enable real time communication with variables within a networked system. As such, and apart from example structures already defined in *VIGO* for trial and familiarization, it will be necessary for such a structure to be defined for the real project. Again, this is quite straightforward, but new users should refer to *VIGO* help files for details about defining a new project.

10.1 Basic Operations

This sub-section provides the opportunity to discuss how to perform some of basic operations used in *Visual VIGO*, such as Moving, Selecting, Undoing, Using and understanding Images, and Printing.

10.1.1 Starting *Visual VIGO*

It should first be ensured that the PC is set to **Small Fonts,** a screen resolution of **1024 x 768** pixels or higher, and a colour resolution of **24 or 32 bits,** in the Windows Control panel – Display properties window. *Visual VIGO* can be started by selecting the application from <Start | Programs | *Visual VIGO*> or from a prepared short cut. When first started, you will be presented with a blank *Work floor* and a set of menu items. However, if you have already worked with *Visual VIGO* before and saved a *Design*, the last one worked upon will immediately be shown. If this file had been closed in *Run mode*, it will be opened again in *Run mode*. In any event, selecting <File | New> will open a new *Visual VIGO* container window together with a blank *Work floor*. It is advisable to choose a name for your design by using the <File | Save As> menu item to save it with a file identity of your choice. If this is not done, Visual VIGO will save it with a default name such as *Design1, 2, 3* etc.

10.1.2 Selecting Components

The selecting of a component is one of the most fundamental activities for a *Visual VIGO* designer. It is necessary for copying, moving, sizing and setting the component properties. Although it an obvious and intuitive action to take, there are few alternative ways in which it can be achieved.

The way to select any single component in *Design mode* is to single click on it with the left-hand mouse button. Dark boundary markers will be shown around the object and enables drag moving or drag sizing to be performed. If components are selected with the right-hand mouse button, then the editing pop up menu will be displayed at the same time to enable copying, rotating etc. to be performed. If the *Component tree* is already displayed, selection of individual components will highlight the name of the component and show the appropriate *Property detail window* for each selection.

An alternative way to make component selections, is to click on the name of the component in the *Component tree*. This will again show the selection on the design screen and highlight the component on the tree.
To select a number of components at the same time, perhaps for the purposes of grouping or moving, select the first one in the normal way and then hold down the shift button while clicking on additional components. The extra components will display gray boundary markers, indicating that they can no longer be resized, but the joint selections can then be moved together. If this multiple selection is then grouped, the Property tree will adjust the structure to show a *group* name containing the included components.

An alternative way of selecting multiple components which are in close proximity, is to hold the Shift key down while holding down the left-hand mouse button, and moving the mouse such as to form a rectangular box around the required selection of components. This will produce gray boundary markers around the composite selection, indicating that the group can be moved together but not resized. While this new group is still selected, additional individual or groups of components can added to the original multiple selections. All these components will have individual gray boundary markers.

To deselect a multiple selection, select another individual component or press the <ESC> key.

10.1.3 Moving Components

Individual or multiple selected components can be moved around and positioned on a screen in a number of different ways.
The easiest way is to position the mouse pointer over an object and hold the left-hand mouse button down. Now drag the selection to the desired position and then release the mouse button. The reference point of the object is shown by the selection mark in the top left hand corner. If <*Snap to Grid*> is switched on, then on release of the mouse button, the reference mark will position itself at the nearest grid point. To override this feature, hold down the <ALT> key whilst positioning. This will enable an object to be accurately positioned at a point other than at a grid mark.
Another convenient way of moving components around a screen is to use the cursor control keys (<arrow> keys). Once a component is selected, movement in a certain direction can be attained by the selecting the appropriate key. Again, if <Snap to Grid> is operating, each press of a key will move the component one grid space at a time. Similarly, if the <ALT> key is activated, accurate positioning outside the grid marks can be achieved. It should be noted that by positioning a component reference point outside the grid using this method, subsequent key controlled movement will move the component by a grid distance, but will not be referenced to the original point. However, the original reference point will be used immediately the mouse is used for movement.

10.1.4 Undo/Redo

Visual VIGO provides powerful facilities to retrace steps taken during the design process. Actions such as performing moving, resizing, revolving etc. to a single or a number of components in turn, can be undone. Furthermore, such actions can be “replayed” (Redo) before a decision is taken to save the current state. However, this facility also applies to the changing of properties, where activities of choosing styles, colours and images can also be undone and redone. The main thing to remember is that if a design is saved, the list of previous actions will be lost. Should an action be taken that requires that *Visual VIGO* needs to perform an automatic save of the design folder, the user will be warned, with the opportunity to reject that action. The easiest way to perform Undo is to use <Ctrl>+<Z>, and <Ctrl>+<Y> for Redo.

10.2 Design Tools and Visualisation Principles

This section discusses aspects and provides hints for laying out a Design.

10.2.1 Visual VIGO Dimensioning

When a *Visual VIGO* window is displayed on a PC screen, it will only cover a certain proportion of the available viewing area. Indeed, even when it is maximized, it is likely that some of the screen will also be occupied by the task bar. One of the powerful features of *Visual VIGO* is that a design made on one PC, having a certain screen size and resolution, can be transferred to another PC with a different screen size and/or resolution, without affecting the general proportions or sizing dimensions of the design. Therefore, when sizing or positioning *Visual VIGO* components within a design, dimensions are expressed in terms of a percentage of the *Visual VIGO* window width. Thus, if a *Work floor* is sized is width = 100% and height = 65%, it will appear to occupy the whole of the *Visual VIGO* window area, no matter what type of PC screen it is displayed on. Alternatively, if the *Work floor* is sized as width = 50% and height = 32.5%, it will appear to occupy one quarter of the displayed *Visual VIGO* window area.

Component dimensions

Component dimensions are also expressed in the same way, in terms of a percentage of *Visual VIGO* window width. Therefore, specifying the dimensions of say an *Image Container* component to be 1.0% x 1.0% will produce a square image having a width and height of one hundredth of the *Visual VIGO* window width.

Component position

The relative positioning of components is also expressed as a percentage of the *Visual VIGO* window width. In this case, position 0, 0 is in the top left hand corner of the window, whilst 100, ~65 would relate to the bottom right hand corner of a maximised window.

This concept is important to understand when considering the “fit to frame” and “zoom” facilities provided by *Visual VIGO*.

10.2.2 Grid

The grid can be regarded as a useful means of defining the requirements of snap spacing and/or providing a visual means of scaling work floors and components. Grid spacing also relates to a percentage of the Visual VIGO window width, as a floating point value. Thus if 10% is chosen as grid spacing, 10 grid dot columns will appear across the maximised window, spaced 10% of window width apart in both directions.

10.2.3 Sizing Components

A useful facility is provided when setting the size of a component. When a component is selected it is shown with boundary markers, which can be used to resize the component in various directions. If one of these boundary markers is clicked upon and held for a moment, the current size in terms of percentage of screen width of the selected component is provided as a hint.

Fit to frame

When importing an image into a component, the designer is provided with the facility of stretching or squeezing the image to fit the component frame. This is possible with all image types defined in the image file selector, i.e. jpg, jpeg, bmp, ico, emf, wmf. Thus, if a component is drawn with a frame size of say 10% x 10% and an image whose original size is 32 x 32 pixels is chosen, then the image will appear not to fit the component frame in being too small. However, if the <Fit to frame> property is chosen, then the original image size will be increased and will now fill the frame. On the other hand, if the original size of the image is 200 x 200 pixels, then not all of this image will be seen inside the frame unless the <Fit to frame> facility is used. By selecting the facility in this case, the image will be reduced to fit the frame.
The importance of this facility cannot be overstated. It means that a wide variety of images can be utilised to fit the chosen component size, rather than having to make the component fit the image. Furthermore, by later resizing the component, the embedded image will resize in the same proportion. It is therefore highly recommended, that apart from special cases, all imported images use the <fit to frame> facility.

Aspect Ratio

When an image is designed, it is saved with a definition of its size in pixels. If an image has been designed to be square, it is said to have an aspect ratio of 1:1. In other words, its width is the same as its length. If such an image is imported into a component frame, which is also square, then stretching the image will still retain the original aspect ratio. However, if either the image or the component frame has a dissimilar aspect ratio, then by fitting the image to the frame, it will appear distorted (morphed). Although this can produce some interesting effects, it is more likely that an image (picture, photograph, drawing, symbol etc.) has been chosen to be presented as the original. Therefore, when choosing an image from the import image browser, note the pixel dimensions in the preview details. If the image does not have the same aspect ratio as the component frame, then the component frame will need to be resized to suit.

Zooming

The zoom facility is not only useful when matching images to component frames or placing components accurately during design time, but also during run time, to focus on a particular part of a process. In all cases, the easiest way to control the zoom is to use the keys <Ctrl> + < I > (zoom in), <Ctrl> + <O> (zoom out) or <Ctrl> + <N> (revert to normal).

Zooming only applies to components and images that have the <fit to frame> property selected. Components are visually structured using vector graphics. This means that when they are enlarged, they do not loose their visual quality. If an image is also structured using vector graphics (e.g. wmf), there will also be no loss of quality if resized or zoomed. However, if the image is of raster (bitmap) origin (e.g. .bmp, .jpeg), then the higher the zoom ratio, the more likely that individual pixels will start to be seen (ragging). This may be of no consequence if zooming as performed on a temporary basis, but the designer should be aware of the resolution and type of images when designing a layout, especially if components are to be drawn quite large, or if there are to be many components displayed on a single screen. It should be realized that the higher the resolution of each image, the higher the memory required to store a design or a screen picture. This also applies to printing a picture of a particular *Work floor*.

Fit to Window

Another useful feature incorporated into *Visual VIGO*, applies to when a user resizes a *Visual VIGO* window in “normalised” view. The window will either display vertical and/or horizontal scroll bars in order to reveal the whole picture, or by using the “Fit to window” facility, will attempt to fit the design into the dimensions of the normalised window, without changing the design’s aspect ratio.

10.3 Working with Images

Visual VIGO is all about associating images and other media with specific functional objects that we call components. As a generic term, we define an Image to be anything that can be seen on a computer screen, such as a drawing, a photograph, a symbol, an icon, clip art etc, or just a colour

There are thousands of images available for you to use, ranging from clip art and icons within your operating system or installed applications, images captured from web pages on the Internet, to specialized industrial images available for purchase from third party suppliers. Alternatively, you can draw your own!

When using the standard set of components provided, you normally have the choice of associating images with them, either as a background or to indicate a change of state. Choosing an appropriate image to convey information to the user of your design is one of the most important aspects of using *Visual VIGO*. For example, you may wish to show a valve changing state. You can do this in a number of ways, by either finding two or three similar images which have different colours, or using images which show a positional change, or a combination of both. Alternatively, you could choose a single image for a valve and change the background colour.

10.3.1 Image Types and Resolution

Since Visual VIGO utilizes imported images within components to convey information to the user, it is important to understand the basics of image resolution, when designing a visualization system.

Images on a PC screen are displayed using pixels (picture elements), being the smallest area that can be “drawn” on the monitor. PC monitors specify their maximum resolution or area in pixels. However, it is the setting of the video adaptor card, which defines the current resolution of the screen. Some common settings include: 640 x 480, 800 x 600, 1024 x 768 and 1152 x 864. It may be realized that the ratio between the parameters of each of these settings is 4:3, which matches the viewable width and height ratio of all but the most specialized monitors.

The size of a pixel depends on the size of the screen and number of pixels that can be resolved within that area. Obviously, the larger the screen and the higher the pixel resolution means that a particular image can be structured more precisely, and will therefore be displayed more clearly. The *Visual VIGO* designer should therefore be aware of these concepts when deciding on a screen layout. Consideration should be given as to whether the design will be used on the same screen resolution as the design. *Visual VIGO* has a special technique for saving the layout of a design in a unified form. This means that if the screen resolution is changed, the percentage of screen area used will stay constant. Is it intended to design a layout, which is larger than the screen, such as to be able to scroll from one end of a factory to another?

When selecting and importing an image into the *Presentation Selector*, the file select window provides a snapshot of the contents of the file under consideration. It also provides information about the dimensions of the image in pixels. In designing a screen layout and setting up user-configured components involving images, it is important to understand the relationship between screen resolution and image resolution. Since *Visual VIGO* has the versatility to “stretch” images to fit within the frame of a component, and to zoom in on a particular area of a *Design*, it is also important to consider how such effects will affect the look of the display to the user.

The system designer needs to be aware of what kind of monitor is to be used in the normal operation of the system, and whether any “plug-in” lap-tops or remotely sited PC’s are likely to be used. One of the features of *Visual VIGO* in particular, and *P-NET* in general, is that a number of visualization stations can be monitoring a process at the same time. All this means is that a layout designed for say a 1024 x 768 monitor, would not be seen at the same definition if subsequently displayed on a 640 x 480 screen, but they will both be seen with the same screen layout.

10.3.2 Transparent Pixels

Some drawing programs provide the facility to define particular image pixels as transparent. This means that when seen on a screen or in a document, such pixels will follow the background colour. If such an image is used in *Visual VIGO* it will also “see through” these pixels. However, if an image is imported where this process has not been performed, *Visual VIGO* provides the facility to define whether a background colour should be seen or become transparent. This switch is provided in image property windows, and performed by analysing the colour of the lower left-hand pixel of the image. If the <Transparent Pixels> check box is true, all pixels with the same colour as the reference pixel will become transparent, and whatever happens to be behind the image will be seen.

.

10.3.3 Image File Types

There are basically two generic types of image. The first is called a Bitmap or Raster graphic image, and the second is called a Vector graphic image. The main difference between them is that with bitmap images, each individual pixel of an image is defined, and the computer (video card) is given instructions for each and every one of them. On the other hand, Vector graphics describe images as a series of mathematical statements. Such formulae define lines and curves as well as colours and position points.

When using bitmaps, consideration needs to be given to the fact that when resizing such an image, there is a clear definition of how many pixels have been used to make up the original image, and depending on the resolution of the original drawing, could make the image look ragged if expanded.

Vector graphics do not suffer from this problem, in that if expanded, it's mathematical definition is adjusted so that length of lines or position of certain points change, but the overall quality of the graphic remains unchanged.

Visual VIGO supports both types of image and recognizes a variety of common file types:

BMP – Bitmap. This is a bitmap image and one of the most common file types used in the Windows environment.

JPG & .JPEG – Joint Photographic Experts Group. The JPEG image file is a compressed form of a bitmap and basically saves file space by embedding instructions to regard a long line of similar pixels in the same way. This is one of the common image formats for images used in web based (HTML) documents, since it is compressed. Expanding or zooming in on such images needs consideration, and if a problem, the uncompressed equivalent should be used.

ICO – Icon. This is also a bitmap but is normally treated as a set size and resolution. However, in *Visual VIGO* these can also be stretched or resized. These are usually constructed using 32 x 32 pixels. They are commonly used for images for toolbar buttons, and if searching in a file manager, it will often show the image next to the file name.

WMF & .EMF– Windows Meta File & Extended Meta File. These are vector type image files. If stretched or resized the quality of the image is not affected. This means that they can be resized with ease without affecting quality resolution.

10.3.4 Fonts

Visual VIGO provides many situations where the designer can choose the type, style and size of a font (scales, text formatting etc.). As with image type and resolution, there are both raster and vector type fonts. It is therefore highly recommended that True Type fonts are used in all but the most special case, since zooming or resizing will not change the quality of such characters.

10.4 Working with Multimedia

The way a design is presented to a user can be significantly enhanced with the use of multimedia.

10.4.1 Sound

One form is sound, which can be derived from an analogue (i.e. *.wav) or digitally synthesised (e.g. .mid) type sound file. As with a static visual image, there are many to chose from within your PC environment or the Internet, or you can make your own. Consider the operation of opening or closing a valve, or starting and stopping a pump. In addition to using a two-state component, which can change colour and show a new image or text, it may be very useful to hear a particular sound during this activity. This could either be in the form of a short sound of the type heard when using Windows “office” type applications, such as a “Click” every time the button is pressed. Alternatively, it is particularly easy to make a microphone derived wave file, using the Sound Recorder utility provided within the “Accessories” of the Windows operating system. In this way, specific warnings or other audible information can be given to the operator, such as “Valve 3 Opening” or “Pump 2 Stopping”. This can be taken further, to give a verbal error message if anything untoward happens to a variable. Such sound enhancements may be even more useful when used with alarm set points, where different sounding audible alarms can be given for a particular type of alarm, without having to set up specialist hardware to do the same thing. One the other hand, a pre-dictated “Tank Four High” may be more appropriate. Such sounds may be arranged to be played once, or repeatedly until the alarm stops or is acknowledged.
Another useful feature within a design might be to re-play some verbal instructions to an operator from an array of appropriately labelled help buttons.

10.4.2 Video

The other form of multimedia is visual, but this form (*.avi) is moving or animated, with or without soundtrack. Not so many of these files exist for immediate use as do sound files within the Windows environment, but again are reasonably easy to produce via a video camera or animation application program. Consider the use of such a form to show the movement of a valve or repositioning of a diverting device. Perhaps an animated clip showing the operator how to perform a particular activity could be opened when activating a Help button on a *Work floor*.

The use of multimedia as an additional presentation property of a component provides the designer with a much-increased range of possibilities when devising a new design to perform a specific task.

10.5 Printing

Visual VIGO provides many versatile facilities for both printing a *Work floor* image on a printer and saving a picture of a *Work floor* as a file for later use in documentation or publishing on the Web.

10.5.1 Print Facilities

In the *Print set up window*, the equivalent absolute size of the picture that will be printed on paper for the selected *Work floor* is shown. This relates to the area of the selected *Work floor* only, and even if this covers the whole screen area, does NOT include the menu or toolbar, or the tray. These dimensions can be shown in centimeters, inches or pixels. The picture to be printed is NOT a direct screen dump of the *Work floor* shown on the screen, but is derived from a “virtual canvas” with a resolution specified when printing. The default print resolution is 96 pixels/inch (approx. 38 pixels/cm). As such, and depending on the size of the selected *Work floor*, will result in a certain amount of memory (3 bytes/pixel) being required for the print process. This requirement is shown in the *Print set up window*. It is worth remembering that capturing a work floor picture from a higher resolution screen, will result in a higher memory usage, but will also produce a more detailed printout.

The captured screen picture can be resized within the paper area by either changing the dimensions in the <height/width> text boxes or within the print preview window by dragging it to the size required or using <main|Fit to paper> menu item to do this automatically. In both cases the original aspect ratio will be retained. Note however, that the picture is now a bitmap, and as such, stretching a small picture to fit a larger paper area, may produce a problem with perceived resolution and require more memory. On the other hand, reducing the picture to paper ratio, will reduce memory, but will also reduce definition.

10.5.2 Saving screen pictures to File

The captured bitmap picture may also be saved to file for use in documents and manuals. The image file type is jpg, which is a compressed form of .bmp. This is also a form used for images on web pages. The user has the option of saving the picture with no compression (100 % compression quality), or can apply compression to suit the purpose. Generally speaking, selecting 50 will reduce the file size by 50%, but as with printing, a compromise between file size and image quality needs to be considered.

11 Making and Developing New Components

This section firstly discusses how new components can be made from standard components and finally provides basic information about what is required to develop new components with a new functionality.

11.1 Making new components

11.1.1 By Changing Properties

Whenever a single property value of a standard component is changed, it could be regarded as a new component. Obviously, if just a small change to a name, colour or font is made to a standard component during the design process, it is unlikely that this would be felt to be worth reproducing as another component. However, if the designer decides to change the size, orientation, scale, image etc. of a standard component, then such a customised component may be worth saving for future use, in this or other designs. All that is required is to ensure that the new component is given a recognisable generic name, e.g. *MyNewBarGraph*, and that any *PhysID* is blank. Now just select it, and copy with the right-hand menu. Open the default or a new *Component box* and paste it into an appropriately named tab. The new size, orientation, scale and image properties are now saved along with the base functionality of the new component. The procedure of clicking on this new *component* in the *component box* and then on the desired position in the *design* will produce as many copies of the new component as required. All the designer has to do is to name each new instance of the component and perhaps set the *PhysID.* Such a new component can be sent to others by attaching the named container *component box* in an email.

11.1.2 By Grouping

The same principle can be applied to a grouping of components, which may be an amalgamation of the same or different types of components. This may be useful if a repeatable "panel" of components is required, or perhaps a facsimile of some kind of instrument involving switches, lights and measurement indication. In this case, the individual components are not integrated, but are contained within a group map, having its own name and hint properties. When saved as a new composite component, the button image used will be a thumbnail of the image of the group within the design. Such a composite component will also be reflected as such within the *Component tree*.

11.1.3 By Layering

Another way of making a useful new component is to use the facility of layering by placing one or more components on top of another. Consider that we want to indicate a particular range from a single analogue variable, by means of a colour change or the change of a pointer image. We first take a *Two-state* component and size it using the assistance of the grid. An "off " image is then chosen, which is to be seen when an analogue variable is within a particular range. Next, make the "on" image transparent. Then set the *Function* properties of the *Two-state* component to define the numerical level (trigger level) for where it is required for the image to change state. Finally, set the *PhysID* to the specific variable for this indicator. It might now be useful to save a copy this initial set-up, especially if other properties of the standard component have been changed such as frame style and colour.

Using a copy of the first component, change the trigger level and the image to suit. Continue this for as many ranges as you wish to indicate. We now want to layer each component one on top of another. At a certain level, the second component will be seen “through” the first, because it has become transparent. As additional layers are added, further discrete ranges of a single variable can be indicated. Following testing, such a multiple state component can be grouped and saved as a new component.

11.1.4 By adding Run Time Help

Design time Help

Standard components that are delivered within the *Visual VIGO* package also contain a *design time* help file. This web style file is incorporated when the component type is being designed, and is intended to assist the designer when using such components within new layouts. Thus, when in Design mode, the designer can select a component, and then press <F1>. This will open a window containing standard help information about the selected component. Furthermore, by selecting a component element from the *Component tree* and pressing <F1>, will display help information about setting a particular set of properties. Thus, by making changes to the properties of a standard component and saving it as a new component, it will continue to incorporate the *design time* help facilities.

Run time Help

In addition to the built-in help for the designer during *design time*, each standard component also incorporates some default *run time* help. Rather than this being associated with guidance as how to set properties etc., the *run time* help focuses on how a particular component type is used. However, a powerful facility is also provided so that the designer can add his own *run time* (User) help file to any component that has been configured for a specific purpose. For example, if a particular standard component has been configured to represent say a pump, then a help file can be prepared such that if the mouse pointer is placed over this component when the design is in *Run mode* and <F1> pressed, that a help viewer will be opened showing the designer’s help file. The text displayed will be specific to this component, and if indeed it is the pump component, can show information about the specification or operation of this particular pump, and if required, include drawings, wiring diagrams, maintenance schedules etc. This will override the use of the default run time help file, although the designer can still choose to include this file by including a hyperlink to “Details”.

The preparation of such a help file for association with such a designer configured component is quite straightforward, and performed when in *Design mode*.

First, a document is prepared, using a standard word processor that provides the facility for documents to be saved in HTML format. This is the same format used to design Web pages. The document can include drawings and pictures in the same way a Web page is prepared. In addition, hypertext links and bookmarks can be incorporated, so that the user can jump to various positions within the displayed document. Indeed, such a document can be structured such that it can be utilised for more than one component. Whatever the chosen structure, such a document will be saved in a folder containing the HTML document and any image files. It may contain additional sub-folders containing other “pages”. The main folder should be saved with a recognisable name, because it is going to be called upon during the next stage of preparation.

Second, while in *design mode*, the component to which this prepared HTML folder is to be related, is selected either in the design or from the *Component tree*. By highlighting the component name, the property editor will show the window detailing the name, hint, size, position and component .dll. By pressing the <Select> button opposite the [Run-time help] label, another list window will open. This will show all help folder names that are already associated with *run time* components. To add the newly prepared help folder, import it using the presented file browser. The name of the folder will appear in the list and will automatically be selected as the *run time* help for this component This user help for the particular component is now ready to use in *Run mode*.

The designer is given flexibility in choosing how his help documents are structured. Apart from ensuring that help files are saved within individual folders, he can choose whether he wishes to contain user (*run time*) help within one composite folder, or to produce a number of smaller individual folders. He can consider providing an external link with a previously prepared technical manual or even an external web site, perhaps provided by the manufacturer of the example pump.

In essence, by associating specialised help with a particular component, produces a new component, which can be used elsewhere in this or other designs. Additional information about producing and incorporating customised help files can be found in the *Visual VIGO* Help System section

11.2 Developing New Components

This section refers to how to obtain further information about developing new components with new functionality, since the scope of this users manual does not extend to the wider aspect of object-oriented programming.

A new component is developed using the Delphi programming language, by producing a new *Visual VIGO* DLL

Programming is object oriented, and based on the *Visual VIGO* class library and the normal Delphi classes. All the functions relating to the *design-mode* are taken care of by the *Visual VIGO* class objects.

Design time help files can also be incorporated during the component development stage.

Programmers wishing to develop new functional components and have experience of Delphi and object oriented programming can contact PROCES-DATA for details about courses and support.

11.3 Procurement of New Components

The policy of PROCES-DATA is one of continuous product development. As such, additional components will be produced, which will be available for purchase. It is envisaged that many of these visual objects will be structured as “function blocks” enabling processes on the network to be embedded within controllers without the need for any programming. Visit the PROCES-DATA web site for up-to-date information.

12 Sample Designs and Tutorials

This section describes the close association between Visual VIGO and VIGO in terms of the use of VIGO workspaces and projects, and some sample Visual VIGO designs.

When VIGO 5.0 is installed, it includes 3 *Workspaces*. A *Workspace* is the means of defining yours or somebody else's PC set up, where the PC in question ('This PC'), has already been included as a node type (VIGO_PC) within a particular VIGO Project. That is to say, which drivers are to be loaded when a specific *Workspace* is selected , what named network/s you are connected to, what the node address of your PC is, and how many masters are on the network, etc. This is done by associating a named *Workspace* with any one of the PC nodes included in any of your *Projects*, by using the 'This PC' selector in the *Workspace* tab of VIGO. You can set up your own *Workspace* PC associations, and/or export or import *Workspaces*, including selected projects, for this or other PC's. See the VIGO user manual for more details. The 3 *Workspaces* delivered are:

- Basic Workspace – This is associated (selected as 'This PC') with *one* of the PC's within the pre-defined Sample Project. One PC ('ThisPC_3930LPT1') includes a P-NET/RS485 driver for parallel port interface module PD 3930 (included in the Evaluation Kit). As far as the P-NET driver is concerned, it assumes that your PC has a node address of 1 of 6 masters. If this is not the case, it is easily changed. If you have the internal card interface module PD 3920 installed, then the second PC ('ThisPC_3920') within the pre-defined Sample Project, should be selected as 'This PC' instead. If you don't have any hardware installed as yet, then the Demo Workspace should be used, to avoid 'driver not loaded' errors. See the VIGO user manual for more details.

- Demo Workspace – This *Workspace* is also associated with a PC, but in the Process Plant project , and as such, no hardware drivers are loaded. However, an IP (Internet Protocol) driver is loaded to enable communication between a sample design and our premises in Denmark. **This is the default association.** This *Workspace* would normally be selected when no hardware is installed and for working with *Projects* and *Designs* that have only been configured for simulation mode (on-line access property not selected) and don't require any P-NET communication outside the PC, e.g. Process Plant Project, Tutorial and Demo Design. As far as the IP driver is concerned, it is already configured with our IP address (PROCES-DATA.com), and it assumes you have dial-up Internet access via your local Internet Service Provider (ISP). The only thing you need to do is to specify your own computer's IP address, by selecting it from a network browser from the IP drivers properties.

- MODBUS Workspace – This Workspace is associated with the PC included in the sample MODBUS project, and which loads an RS232 type driver for communication with a PLC having a ModBus protocol interface.

When VIGO 5.0 is installed, it includes 4 *Projects*:

- Sample project – This includes some standard PD module types for use with the Evaluation Kit, and for other demonstration and training activities. It requires to operate within a *Workspace* having a loaded PC to P-NET interface driver, e.g. Basic Workspace.

- Process Plant project – includes some PD module types in simulation mode, where certain module variables have been gathered into named *groups* containing named *aliases* for specific use with the Visual VIGO Process Plant design. It does not require any network drivers or hardware.

- PROCES-DATA project – This includes the same structure as that of a network of some standard modules actually installed at our headquarters in Denmark. Some accessible module variables have been gathered into named *groups* containing named *aliases,* for specific use with the Visual VIGO PROCES-DATA design. Requires a *Workspace* having a loaded IP driver for communication via the Internet, e.g. Demo Workspace.

- ModBus project – a means of interfacing with a standard PLC node having a Modbus comms. port. It requires a *Workspace* having a loaded Modbus RS232 driver, e.g. MODBUS workspace.

VIGO 5.0 also includes the *Visual VIGO* application. *Visual VIGO* includes a sample *design* called Visual VIGO Demo. This is the default design when Visual VIGO is first opened, and consists of three parts:

- Process Plant – a simulated example of an imaginary plant, including tanks, valves, flow meters, pipe-work etc. It uses the Process Plant project MIB in VIGO.

- PROCES-DATA – a pictorial view of one of our buildings at the PROCES-DATA headquarters in Denmark. Its purpose is to show an ability to connect to remotely located live measurements and controls via the Internet. It uses the PROCES-DATA project MIB in VIGO.

- PROCES-DATA – a pictorial view of one of our buildings at the PROCES-DATA headquarters in Denmark. Its purpose is to show an ability to connect to remotely located live measurements and controls via the Internet. It uses the PROCES-DATA project MIB in VIGO.

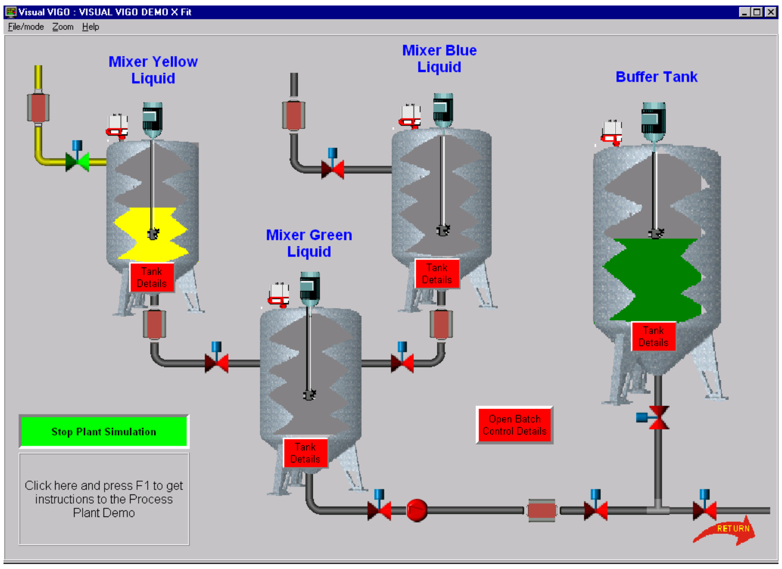
13 Visual VIGO Runtime help

13.1 Visual VIGO main run time help

13.1.1 General Features

Visual VIGO is a Windows based SCADA (Supervisory Control and Data Acquisition) application program, used to visualise and operate process plants, factory automation, building management systems, etc. on a PC. The PC can be connected on-line locally, or remotely via a LAN, direct MODEM access or Internet connection, through the *VIGO* field bus management system and the P-NET fieldbus.

A process plant is visualised by designing one or more *Work floors*, where machines, equipment and instruments can be placed. The operator is able to “walk” through the factory by opening “doors” to the different *Work floors* and departments of the Factory. The different machines and instruments can be opened for studying internal details.



Visual VIGO is based on an object-oriented principle, where *Visual VIGO components* can contain other *Visual VIGO components*, each of which can contain other *components* etc. *Visual VIGO components* are placed on the *Work floors*, and can be used to illustrate a complete working floor, a machine, a detail of a machine, an actuator, a sensor, a control function, a value, a set point, a data recorder etc.

The way a *Visual VIGO* component is presented to the user involves the use of images, multimedia items and help information. *Visual VIGO components* can take different states. These states can be represented by using a set of chosen images, and if required, include sounds and visual effects. The images could originate from a picture taken with a camera, be generated by any standard drawing program or computed by *Visual VIGO components*. Multi-media effects are based on standard .avi and .wav files generated by the user or chosen from a variety of prepared sources. If required, Help information can be generated by the designer using standard text preparation facilities and associated with a particular component.

13.2 Terminology

Throughout this manual, certain common terms are used, which have a specific meaning in the context of a particular description. The following paragraphs provide a short definition of such words, phrases or functions.

13.2.1 Design

"Design" is the term for a *Visual VIGO* set-up, made to interact with a control system. A design is saved within a single packed file, containing all necessary data for performing the functionality.

13.2.2 Component

"Component" is a term for an object that can be within a *Work floor*. The Components are normally visible and used to indicate the current states of the control system. Each base Component is created as an individual piece of software compiled separately as a DLL.

13.2.3 Work floor

A "Work floor" is a component, which has a basic surface that can hold other components. A design always contains one **main** work floor, onto which all other components are placed. Additional Work floor components can be added in a chosen hierarchical structure. This means that one floor can be opened from another floor and so on, to unlimited levels.

13.3 The Visual VIGO Container

Visual VIGO is a modular system, built up using a *Visual VIGO container* and a number of independent *Visual VIGO components*. The *Visual VIGO container* is physically located in *Visual VIGO.EXE,* some DLLs (Dynamic Link Library) and some images etc. *Visual VIGO components* typically consist of a DLL, together with one or more images, and a help file.

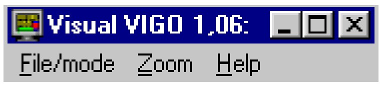
The *Visual VIGO container* can be executed in two modes, *Run mode* and *Design mode*.
In *Run mode,* the main task of the *Visual VIGO container* is to open or close a *Visual VIGO* design and to perform a few service tasks for the components. The most important service task is to activate the components periodically. Most of the functionality of the system is programmed into the included components.

In *Design mode,* the task of the *Visual VIGO container* is to receive edit commands from the keyboard and mouse, to move and copy components, edit properties, undo, redo etc. The components have a number of properties, including size and position. During the edit phase the components receive new property values, and by this means, their appearance and position can be changed.

Access to any edit facilities depends on the VIGO access settings and whether *Visual VIGO* is in *Run mode* or *Design mode*.

13.3.1 Main menu description

The figure shown to the right shows the main menu, as it appears when *Visual VIGO* is in *Run mode*.



File/Mode

The File/mode menu contains the following items:

Open
This function is used to open a previous *Visual VIGO* design. The design is selected from a standard file browser, from where the previously named file is selected. If another design is already open, it will first be closed, after determining whether changes have been made.
If the design is changed but not saved the designer will get the opportunity to save it before the new design is opened.

Close
This function is used to close the current *Visual VIGO* design. If the design has been changed since it was last saved, the user will be asked whether he wishes to save it or not.

Print
This item opens a window giving screen image information about a complete design or a selected *Work floor*. It also provides facilities to size, save as a .jpg image file, to preview, to set up a printer and to activate a print. Greater detail is given in section (How to use Visual VIGO)

Design
This menu item is used to change between *Run mode* and *Design mode*. The initial state of the menu is that this item is ticked when a new design is created, meaning *Visual VIGO* is in *Design mode*, where editing of the new *Visual VIGO* design can take place. To change the mode, just click once on this menu item. The state of the current mode is saved along with the design, and will be restored when the design is re-opened. Toggling between Design and Run modes can also be achieved from the keyboard, by pressing <Alt> + <F><D>

Zoom

This menu item enables the size of the screen image of a design to be expanded or contracted to suit run mode activities. The zoom facility is useful during run time, to focus on a particular part of a process. In all cases, the easiest way to control the zoom is to use the keys <Ctrl> + <I> (zoom in), <Ctrl> + <O> (zoom out) or <Ctrl> + <N> (revert to normal).

In <Ctrl>+<I>
Makes the screen image larger in order to focus in on a particular area of the design. When the design becomes larger than the screen, scroll bars will appear. A particular area can be centralised by using the scroll bar controls or the cursor control keys.

Out <Ctrl>+<I>
Produces the opposite effect to In.

Normal <Ctrl>+<N>
Resets the screen image to the original default size.

Fit to window <Ctrl>+<W>
Resizes the design to fit into the currently displayed area of the window.

Help
This Help menu contains the following items:

Visual VIGO
This item provides information about general *VIGO* functionality. Leaving the mouse cursor over a component and pressing ”F1” opens help information about an individually *Visual VIGO* component.

The following sections are included in the *Visual VIGO* run mode main help system:

- Introduction: Explains the general use of *Visual VIGO.*
- Terminology: Describes the various terms used in the Visual VIGO help system.
- Help System: Contains a detailed description of the help system.
- How to use Visual VIGO: Description of how to operate Visual VIGO.

Main user run time help
Opens the actual help file of the main work floor linked to the design.

About Visual VIGO
Opens a window not only providing details about the version number of the application, but also shows the number of components that are allowed in a design without an additional licence. It also shows a count of the number of components currently being used.

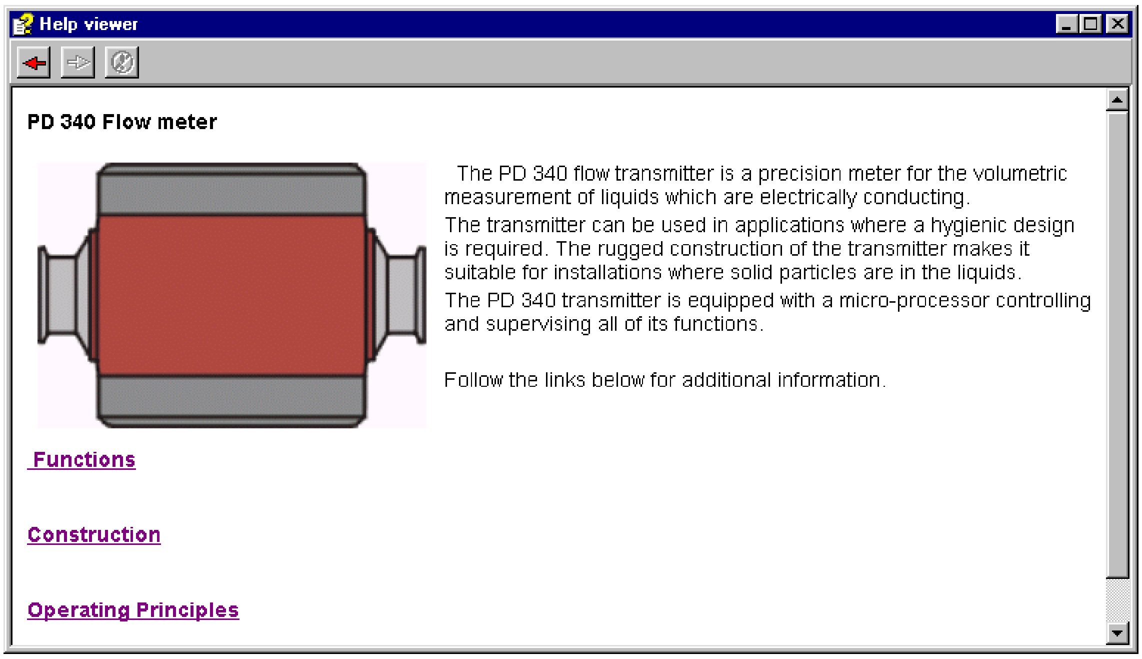
The *Visual VIGO* Help System

Visual VIGO offers a powerful *Run mode* help system to provide context help to the user.

Run mode help

In run mode, generic help on component configuration would not be required. However, in this mode, the highly useful facility is provided for the designer to embed customised help within certain key objects within the operational design. Since no Component tree or component configuration is required in this mode, the mouse is the means for determining which help file is displayed within the *Help viewer*. Positioning the mouse over a particular object and pressing <F1> launches the *Help viewer* and displays a help page related to that object.

Visual VIGO Help viewer



PD 340 Flow meter

The PD 340 flow transmitter is a precision meter for the volumetric measurement of liquids which are electrically conducting.
The transmitter can be used in applications where a hygienic design is required. The rugged construction of the transmitter makes it suitable for installations where solid particles are in the liquids.
The PD 340 transmitter is equipped with a micro-processor controlling and supervising all of its functions.

Follow the links below for additional information.

Functions

Construction

Operating Principles

In *Run mode* the *Help viewer* displays help pages associated with selected objects. The *Help viewer* displays the contents of the html documents imported into the help file selector within the design. Since it deals solely with html files consisting of text, images, hyperlinks and URL's, it can be regarded as having a number of the same features as found in a familiar Internet Browser such as Internet Explorer or Netscape. Indeed, if an Internet World Wide Web URL is embedded into a help page and is clicked upon, the *Help viewer* will display the defined web page, assuming dial up networking and a modem is connected to the PC.

The viewer can be resized, minimised and maximised. It also has a control for stepping forwards and backwards over previously displayed pages. Other features can be found by right clicking on the display where a menu will appear to perform such functions as printing, copying, saving pages as shortcuts etc.

13.4 How to use *Visual VIGO*

Visual VIGO has been designed to be easy and intuitive to use. However, due to its great versatility and power, there are a few operations and concepts that may not be familiar to new users. This section is designed to describe how to utilize these to the greatest effect.

An important aspect to keep in mind when using *Visual VIGO*, is that is closely bound to *VIGO* – The Fieldbus Management System. It is *VIGO* that provides all the mechanisms to define and enable real time communication with variables within a networked system. As such, and apart from example structures already defined in *VIGO* for trial and familiarization, it will be necessary for such a structure to be defined for the real project. Again, this is quite straightforward, but new users should refer to *VIGO* help files for details about defining a new project.

13.4.1 Basic Operations

This sub-section provides the opportunity to discuss how to perform some of basic operations used in *Visual VIGO*.

Starting Visual VIGO

It should first be ensured that the PC is set to **Small Fonts** in the Windows Control panel – Display properties window. *Visual VIGO* can be started by selecting the application from <Start | Programs | *Visual VIGO*> or from a prepared short cut. When first started, you will be presented with a blank *Work floor* and a set of menu items. However, if you have already worked with *Visual VIGO* before and saved a *Design*, the last one worked upon will immediately be shown. If this file had been closed in *Run mode*, it will be opened again in *Run mode*. In any event, if no design is opened and <File | New> is selected while <Design> is ticked, a new *Visual VIGO* container window together with a blank *Work floor* will be launched.

Printing

Visual VIGO provides many versatile facilities for both printing a *Work floor* image on a printer and saving a picture of a *Work floor* as a file for later use in documentation or publishing on the Web.

Print Facilities

In the *Print set up window*, the equivalent absolute size of the picture that will be printed on paper for the selected *Work floor* is shown. This relates to the area of the selected *Work floor* only, and even if this covers the whole screen area, does NOT include the menu or toolbar, or the tray. These dimensions can be shown in centimeters, inches or pixels. The picture to be printed is NOT a direct screen dump of the *Work floor* shown on the screen, but is derived from a “virtual canvas” with a resolution specified when printing. The default print resolution is 96 pixels/inch (approx. 38 pixels/cm). As such, and depending on the size of the selected work floor, will result in a certain amount of memory (3 bytes/pixel) being required for the print process. This requirement is shown in the *Print set up window*. It is worth remembering that capturing a work floor picture from a higher resolution screen, will result in a higher memory usage, but will also produce a more detailed printout.

The captured screen picture can be resized within the paper area by either changing the dimensions in the height/width text boxes or within the print preview window by dragging it to the size required. In both cases, the original aspect ratio will be respected. Note however, that the picture is now a bitmap, and as such, stretching a small picture to fit a larger paper area, may produce a problem with perceived resolution and require more memory. On the other hand, reducing the picture to paper ratio, will reduce memory, but will also reduce definition.

Saving screen pictures to File

The captured bitmap picture may also be saved to file for use in documents and manuals. The image file type is jpg, which is a compressed form of .bmp. This is also a form used for images on web pages. The user has the option of saving the picture with no compression (100 % compression quality), or can apply compression to suit the purpose. Generally speaking, selecting 50 will reduce the file size by 50%, but as with printing, a compromise between file size and image quality needs to be considered.