**FRender gadget**

## Gadget view and pins functionality

A computer screen shot of a white rectangular object with blue lines

Description automatically generated

The gadget has one input pin receiving frames (green one), upper output pin for data about mouse activities output, lower output pin for viewed image output as video frame and control pin on down side.

## How rendering on display is organized

The frame arrived on input pin is going through special analysis and processing sequence.

The first step is input frame analysis for control information. If frames with such information are found, they will be processed (look “FRender Commands”).

If attributes of input frame have strings “x=<X\_coordinate>;” and “y=<Y\_coordinate>”, the rendering of following images will be done with shift [X,coordinate>,<Y\_coordinate>].

After that gadget tries to extract video frames from input frame. If the input frame is a separate video frame, this frame will be selected for rendering. If the input frame is container, the first found frame will be used for rendering. Keep in mind that the last frame added to container will be the first found.

If the video frame is not found, no new rendering process will be initiated.

If the video frame is found, new data for rendering will be prepared and message for image and graphics redrawing will be sent to Windows.

Keep in mind, that not all images and graphics will be obligatory drawn on display, it’s CPU loading dependent, some images may be skipped, but it does not have influence on processing results obtaining, i.e., results view and result data stream are independent. Usually, when the frame rate is less than 10-15 frames per second the all video frames and graphics will be shown.

## FRender commands going through main input

Commands for FRender are passed through the main input pin and identified by frame type and content.

Command frame type is text. In most cases this frame should be not inside input container frame, but there are exceptions.

Notes:

* Labels are case sensitive.
* Separators are shown as commas (,), but space( ), tab(\t), semicolon (;), underscore (\_) and opening parenthesis ( ‘(‘ ) could be used.

There is a list of commands in separate text frame.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| # | Frame type | Label | Content | Description |
| 1 | Text | Scale&Units | <scalar scale as double>  <,<Units as string>  <,<ScaleX as double>  <,<ScaleY as double>  <,<Conjugate Scale (0|1)  >>>> | The main scale is scalar (first parameter).  If Units presented, than on view will be presented measured values with shown units, otherwise pixels will be used.  If units presented the additional 3 parameters will be decoded if existing:  XScale, YScale and Conjugate flag (if Y is inverted, i.e. from top to down) |
| 2 | Text | SetWndHandle | Window handle as string | Handle is number in decimal or hex with 0x prefix presentation. |
| 3 | Text | Resize |  | This is instruction to rendering window to adjust size after host window is changed. |
| 4 | Text | **Not “Resize”** | <wh=<host window handle>>  <x=<xleft>  && y=<ytop>  && w=< width>  && h=<height>> | Assignment of new host window handle and/or rendering position inside host window.  If wh defined, new host will be assigned.  If x and y and w and h defined, new position for rendering in host window will be assigned |

There is list of commands which could be separate or inside container frame:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| # | Frame type | Label | Content | Description |
| 1 | Text | SetCenter | Xc=<X of center>; Yc=<Y of center>; | If Xc and Yc will be found in content, view center of current (existed in container) and following presentations will be in assigned point. |
| 2 | Text | SaveImage | <<# of images>, ><Directory for saving> | After this command the images will be rendered to FRender window and saved to file. There are 2 parameters: number of images for saving and directory for saving.  If a number of images is presented, this number of input images will be saved, otherwise only one image will be saved.  If a directory is presented, this directory will be used for image saving.  Images will be saved as is, i.e., what is shown in FRender window will be saved.  Images will be saved as BMP with name <ImageDir>/<Prefix><TimeStamp><Suffix>.bmp.  About prefix and suffix look following commands |
| 3 | Text | ImageSavePrefix | Prefix for Image file name or “<label>” | This is some text which will be used as <Prefix> in file name on image saving.  If content is string “<label>”, then label of video frame will be used as prefix. Word *label* is not case sensitive and if video frame label consists of full path, the only file name without directories, drive letter and extension (for example, “.bmp”)will be used as prefix. |
| 4 | Text | ImageSaveSuffix | Suffix for Image file name | This is some text which will be used as <Suffix> in file name on image saving |
|  |  |  |  |  |

## FRender commands going through control input.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| # | Frame type | Label | Content | Description |
| 1 | Text | SetCenter | Xc=<X of center>; Yc=<Y of center>; | If Xc and Yc will be found in content, view center of current (existed in container) and following presentations will be in assigned point. |
| 2 | Text |  | Get hTargetWindow | This command returns handle of target window (where rendering performed) on control pin output in format 0x%X |
| 3 | Text |  | Reset | This command clears rendering window content |
| 4 | Text |  | Get Outimage(<number of images>) | This command leads to viewed image output to second output pin as video frame in format RGB (24 bits per pixel). One image per redrawing will be sent. For example, "set outimage(1)" will lead to one image output when next window redrawing will be done |

## Frames attributes and their influence on graphics view

There are several types of input frames influenced on graphics drawing:

* Figure frame
  + One point
  + Line (2 points)
  + Rectangle (4 points)
  + Random connected points
* Text frame
* Rectangle frame

Frame attributes are processed as property kit (look FXPropertyKit class).

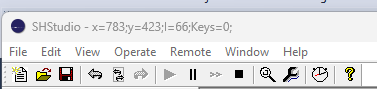
The following table shows possible rendering attributes.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| # | Property | Applied to | Value | Description and Samples |
| 1 | color | All types | String holding decimal, binary or hex number | Value = (Blue<<16) + (Green<<8) + Red  8 bits per color  **color=0;** - black color  **color=255;** or **color=0xff;-** red color  **color=0xffff00;** - cyan color |
| 2 | Sz | Text  Figure (point) | Text size in Pt  Cross side on point view in pixels | **Sz=14;** default value is 12  **Sz=5;** - cross side will be 5 pixels; default value is 1 (cross size will be 3x3 pixels) |
| 3 | thickness | Figure  Rectangle | Line thickness in pixels | **thickness=3;** default value is 1 |
| 4 | style | Figure  Rectangle | Draw style | **style=2;** default value is 0 (solid)  Available styles: 0(solid), 1 (dash ----), 2(dot ….), 3 (dash-dot .\_.\_.\_.\_),  4 (dash-dot-dot \_..\_..\_..) |
| 5 | back | Text | Background color for text as hexadecimal without prefix | Coding as in **color.** If **back** is present then text will be drawn on appointed background, if not – background will be transparent. |
| 6 | x | Text | X position of text on image | **X=100;**  If x and y are present in attributes, text frame content will be shown ([x,y] is coordinates of left top point of text). |
| 7 | y | Text | X position of text on image | **Y=200;**  If x and y are present in attributes, text frame content will be shown ([x,y] is coordinates of left top point of text). |
| 8 | message | Text | Content string | **message=OK;**  If content of text frame is empty, message content will be shown. |

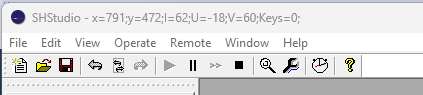
## Manual view control and Setup dialog

The left button click on image leads to pixel under cursor parameters showing.

Pixel parameters will be added to caption of main window. For monochrome image the pixel coordinates (x and y) and intensity (I) will be shown.



For color image the color components (U and V) will be added to caption.

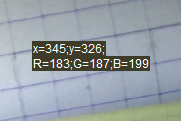
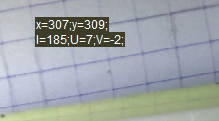


When ViewRGB selection in setup dialog is true, information about color will be shown in RGB form.

A screenshot of a computer program

Description automatically generated with low confidence

If keyboard ‘I’ is pressed when left button pressed, then information about pixel under cursor will be shown directly on image (format dependent on ViewRGB flag in setup).



Second left button pressed when ‘I’ is pressed, then pixel info on image view will be switched off.

If cursor goes out of image, then pixel info on image view will be switched off.

If keyboard ‘F’ is pressed when left button pressed, then ViewGRB flag will be toggled.

‘Keys’ value is bit mask shows what keyboard keys are pressed:

Bit 0 is 1 when Control is pressed (left and/or right).

Bit 1 is 1 when Shift is pressed (left and/or right).

Bit 2 is 1 when Left Control is pressed.

Bit 3 is 1 when Right Control is pressed.

Bit 4 is 1 when Left Shift is pressed.

Bit 5 is 1 when Right Shift is pressed.

Setup dialog shown on following image.

A screenshot of a computer program

Description automatically generated with medium confidence

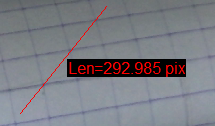
**Scale** allows change scale of displaying

* x1 (image pixel to pixel)
* x2 (image pixel to square 2x2 pixels)
* x4 (image pixel to square 4x4 pixels)
* x8 (image pixel to square 8x8 pixels)
* x16 (image pixel to square 16x16 pixels)
* Fit Window

**Monochrome** option allows switch color-monochrome type of displaying. Monochrome displaying takes less processor resources.

If **LineSelection** is not false, line will be drawn by left mouse button (line begin will be in point where left mouse button is pressed, line end will be in point where button was released). Two operations will be done:

1. Line will be shown on image with printed line length.



1. Text frame will be sent to the gadget output. This frame will consist of line begin and end coordinates, Keys status and line length.



If **RectSelection** is true, the rectangle can be marked on image by right mouse button. When this button is released the rectangle data will be sent to the output pin.



**CutOverlap** parameter is not in using now (it’s for several images overlap).

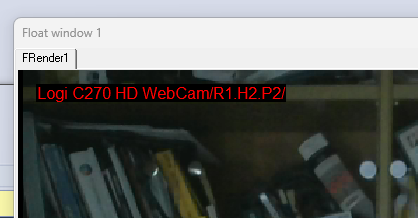
**ViewRGB** parameter controls color parameters presentation as texts.

**RenderId** B is not in use in the current version.

**hTargetWindow** is a Windows handle for image and graphics rendering. If this value is zero, the assigned by default window will be used. User can put to this field existing handle of any window on desktop (this done is for experiments only). Usually, UI application assigns or reassigns this parameter.

**ObjectSelection** is not in use in current version.

**ShowLabel** allows image label presentation on image (if enabled, following label will be shown in left top image corner).



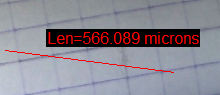
**ScaleUnits** is a field for image scale and measurement units entry. Format is

<Scale in units per pixel><,<units as string>>

This parameter is used for length presentation. For content is



Length will be presented as



Units part is not mandatory. If it’s not present, the word “pix” will be shown as units.

**LineSelection** field has several options:



When units are ‘um’ or ‘microns’ the measured length could be viewed in microns, in tenths (1 tenths = 2.54 microns) or in both forms.

When units are ‘mm’ the measured length could be viewed in mm, in tenths (1 tenths = 2.54 microns) or in both forms.

If keyboard ‘L’ is pressed when left button pressed the **LineSelection** field will be switched.

**SaveImagesDir**  shows where images will be saved. Saving can be done by command only, look p.2 in this document.

## Interactive video object ROI editing

FRender gadget can be used for video object positions editing in TVObjects gadget. Connection between FRender output pin and TVObjects input pin 4 should be done for this functionality using (blue line on following picture).

A diagram of a computer

Description automatically generated

When cursor moving over ROI edge of a video object, the cursor changes form to resize or move arrows.



The real view of these arrows is dependent on specific computer settings.

When cursor has one of these forms, do press CTRL button on keyboard, press mouse left button, move mouse to arrows direction and release mouse left button. Video object ROI will be changed.

Don’t forget to save the graph with new video object parameters.

## Images with graphics output as video frame

When command “set outimage(<# of images>) received on control pin, Frender will send next <# of images> pictures to lower output on each of rendering cycles.

For example, Frender1 did send to Frender2 image when zoom **was** 4, the original image with fitted to window zoom is presented on Frender1.

