The Woolz Internet Image Protocol (IIP) Server (Version 1.1.9)

Zsult Husz [*] Bill Hill [†]

November 18, 2016

Abstract

The Woolz Internet Imaging Protocol (IIP) server (WlzIIPSrv) extends the IIP to allow views of 3D Woolz objects with multiple components. This manual describes these extensions for both users and developers of the WlzIIPSrv.

[*]Supported by NIH under grant #1R01MH070370-01A2.
[†]bill.hill@igmm.ed.ac.uk

Contents

1 Introduction

The Internet Imaging Protocol IIP provides quick access to large image databases allowing large images to be viewed remotely. It makes use of tile-based delivery so that panning and zooming within large images is possible without needing to download the entire image. The computational requirements of the client machine for viewing are low, allowing viewers to be implemented within web browsers. However, it is limited to 2D images. In biomedical imaging (and many other fields) 3D images are routinely used. A set of extensions to the IIP server was developed within the Edinburgh Mouse Atlas Project [EMA] to allow the remote visualization of sections through large 3D objects. These extensions make use of many of the features of Woolz, an image processing system with it's own efficient image representation [WOO].

This document assumes understanding of the IIP [4].

For clarity, in the rest of this document IIPSrv will refer to the original IIP server, with code version 0.9.7 [Pillay and Pitzalis], WlzIIPSrv will refer to the extended server, while IIP will point to the protocol standard of [4].

The coordinate system convention used in this document is shown in Figure 1. The object coordinate is defined by the object and the section coordinates result after sectioning. Then, the translated section coordinates with the section's minimum at $(0, 0)$ are the display coordinates. Further, the section is divided into non-overlapping tiles covering the whole section. Tiles are numbered with 0, 1, 2, etc., with the 0^{th} coordinates matching the display coordinates. The and 1^{st},2^{nd}, etc. tiles continue from left to right and top to bottom. A client can visualise a section with reduced viewing area. This has the view coordinates and is client dependent.

2 Extension overview

In addition to the IIPSrv commands, a WlzIIPSrv command specifies an object, sets the viewing section parameters and requests image or meta data of the object. Tile request are similar to the TIFF requests, however for a Woolz object the resolution number used in **JTL/TIL** is ignored.

The extended command list that sets query parameters are summarised in Table 1, while Table 2 shows the new objects that may be queried with the **OBJ** command.

Command	Purpose	Syntax
CVT	Request an image to be returned as a composed image. CVT accepts JPG, PNG and WLZ format requests.	CVT=*format*
DST	Specify the distance of the sectioning plane.	DST=*dis*
FPL	Specify a viewing section through fitting a plane to the given points or compound indices.	FPL=*fspec[,fcpec]** where *fspec = I|(X,Y,Z)*
FXP	Specify the fixed point of the viewing section rotation.	FXP=*X,Y,Z*
FXT	Specify the second fixed point of the viewing section rotation.	FXT=*X,Y,Z*
MAP	Define a colour or grey value mapping.	MAP= *mspec[,mspec[,mspec[,mspec]]]* where *mspec*=*t,il,iu,ol,ou,p0,p1*
MOD	Specify the projection mode.	MOD=*mode*
PAB	Specify the 3D query point absolute in the object coordinate.	PAB=*X,Y,Z*
PIT	Specify the pitch angle of the sectioning rotation.	PIT=*angle*
PRL	Specify the 2D query point relative in tile or display or tile coordinate.	PRL=*T,X,Y*
PTL	Retrieve a tile as a PNG image.	PTL=*res,tile*
DPT	Specify the rendering depth.	DPT=*depth*
RMD	Specify the rendering mode.	RMD=*mode*
ROL	Specify the roll angle of the sectioning rotation.	ROL=*angle*
SCL	Specify the scale used in the sectioning transformation.	SCL=*scale*
SEL	Specify a component of a compound object to be displayed and its colour. See 2.4.	SEL=*E,R,G,B,A*
UPV	Specify the up vector for the **UP_IS_UP** mode.	UPV=*X,Y,Z*
WLZ	Specify the Woolz object.	WLZ=*path*
YAW	Specify the yaw angle of the sectioning rotation.	YAW=*angle*

Table 1: Extended command overview

Object	Purpose
IIP-server	Identify if WLZ-IIP is running.
Max-size	The size of the section.
Tile-size	The size of a tile.
Wlz-true-voxel-size	The voxel size of the object.
Wlz-volume	The volume of the object or first selection (if it exists).
Wlz-n-component	The number of components in a compound object.
Wlz-distance-range	The range of the sectioning plane distance.
Wlz-sectioning-angles	The pitch, yaw and roll angles of of the sectioning plane.
Wlz-coordinate-3D	The 3D coordinates defined in 2D by the **PRL** command.
Wlz-grey-stats	Simple statistics of the image values of the object or first selection (if it exists).
Wlz-grey-value	The grey or RGB value of a point specified either the **PRL** or the **PAB** commands.
Wlz-transformed-coordinate-3d	The display coordinates and displacement from the sectioning plane of a 3D point defined by **PAB**.
Wlz-foreground-objects	The components of the compound object that 2D/3D query point is a foreground.

Table 2: Extended object overview

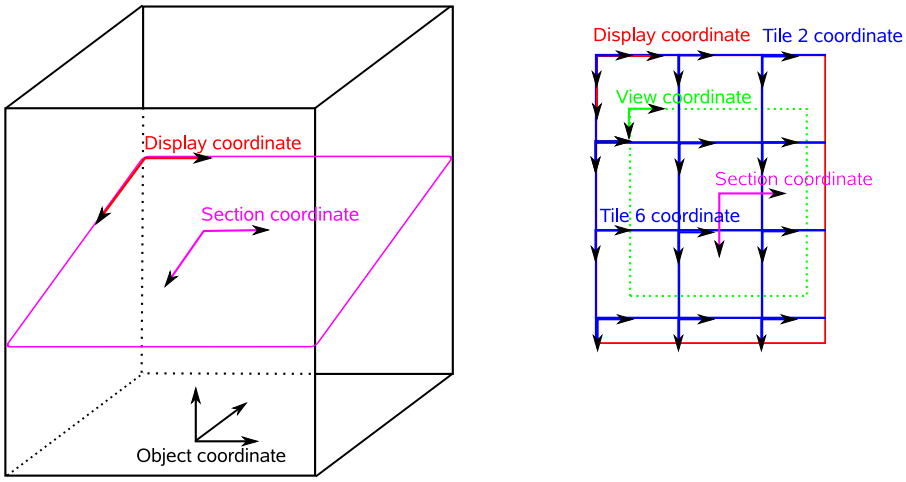

Figure 1: Coordinate systems

2.1 Woolz object specification

WLZ	**Purpose**	Specify the Woolz object. It is similar to the **FIF** command, however instead of loading Pyramidal Tiled TIFF images, it specifies a Woolz object. Compered with **FIF**, the Woolz object is not loaded until a later operation requires this. The Woolz objects are cached. 3D domain and value objects, and compound objects with 3D domain or value components are supported. For a compound object the domain of the first object must include all other object domains. If file name ends with `.gz`, gzipped Woolz object is expected.
	Syntax	`WLZ=`*`path`*
	Input Parameters	`PATH` *`path`* Full path of the Woolz object.
	Response	**WLZ** returns nothing when accompanied by other commands
	Example	⇒`WLZ=/home/bill/pics/lobster3d.wlz`
	Notes	Equivalent if **FIF** [4, p.25]

2.2 Setting sectioning and query parameters

The following commands are optional and can be used in any combination.

DST	**Purpose**	Specify the distance of the sectioning plane.
	Syntax	`DST=dis`
	Input Parameters	`FLOAT dis` The distance to the sectioning plane.
	Response	none
	Example	⇒`DST=12.5`
	Default value	`0.0`
FPL	**Purpose**	Specify the view by fitting a plane to given points or through components of a compound object.

	Syntax	`FPL=fspec[,fcpec]*`
		`where`
		`fspec = I|(X,Y,Z)`

	Input Parameters	`UINT I`
		Index of a compound object component
		`FLOAT X`
		Point x coordinate
		`FLOAT Y`
		Point y coordinate
		`FLOAT Z`
		Point z coordinate

	Response	none
	Example	⇒`FXP=(120,20,15),(567,89,45),(339,87,62)`
	Default value	`Plane through (0,0,0) with PIT=0.0 and YAW=0.0`
FXP	**Purpose**	Specify the view rotation fixed point in the object coordinate system.
	Syntax	`FXP=X,Y,Z`

	Input Parameters	`FLOAT X`
		The x coordinate
		`FLOAT Y`
		The y coordinate
		`FLOAT Z`
		The z coordinate

	Response	none
	Example	⇒`FXP=10.5,20,15.0`
	Default value	`0.0,0.0,0.0`

FXT	**Purpose**	Specify the second fixed point in the object coordinate system of the viewing section rotation used only with `MOD=FIXED_LINE`.
	Syntax	`FXT=`*`X`*`,`*`Y`*`,`*`Z`*
	Input Parameters	`FLOAT` *`X`* The x coordinate `FLOAT` *`Y`* The y coordinate `FLOAT` *`Z`* The z coordinate
	Response	none
	Example	⇒`FXT=30,-20.2,15.0`
	Default value	`0.0,0.0,0.0`

MAP	**Purpose**	Defines a colour or grey value mapping. Grey values outside the mapped region are clamped to the minimum and maximum output colour componentor grey values. There may be 1,2,3 or 4 mapping specifications given; with a single mapping specification meaning a grey value mapping, two specifications a grey with alpha mapping, three an RGB mapping and four an RGBα mapping.
	Syntax	`MAP=`*`mspec[,mspec[,mspec[,mspec]]]`* where *`mspec`*`=t,il,iu,ol,ou,p0,p1`
	Input Parameters	*`t`* $\in$ *`IDENTITY	LINEAR	GAMMA	SIGMOID`* The mapping function `FLOAT` *`il`* Input lower grey value `FLOAT` *`il`* Input upper grey value `FLOAT` *`ol`* Output lower colour component or grey value `FLOAT` *`ol`* Output upper colour component or grey value `FLOAT` *`p0`* Gamma ($\gamma$)or first sigmoid parameter ($\mu$) `FLOAT` *`p1`* Second sigmoid parameter (σ)
	Response	none
	Example	$\Rightarrow$`MAP=LINEAR,0,4095,0,255` Input grey values in the range 0-4095 are mapped to output values with the range 0-255
	Default value	`IDENTITY`

MOD	**Purpose**	Specify projection mode.
	Syntax	`MOD=`*`mode`*
	Input Parameters	*mode* ∈ `STATUE | UP_IS_UP | FIXED_LINE | ZERO_ZETA | ZETA` The projection mode
	Response	none
	Example	⇒`MOD=FIXED_LINE`
	Default value	`UP_IS_UP`
PIT	**Purpose**	Specify the pitch angle of the sectioning rotation
	Syntax	`PIT=`*`angle`*
	Input Parameters	`FLOAT` *angle* rotation angle in degrees
	Response	none
	Example	⇒`PIT=180.0`
	Range	`0.0-180.0`
	Default value	`0.0`
PAB	**Purpose**	Specify a 3D query point absolute in the object coordinate system.
	Syntax	`PAB=`*`X`*`,`*`Y`*`,`*`Z`
	Input Parameters	`FLOAT` *X* The x coordinate `FLOAT` *Y* The y coordinate `FLOAT` *Z* The z coordinate
	Notes	If both **PRL** and **PAB** are specified then **PRL** has priority of feature queries
	Response	none
	Example	⇒`PAB=200,-50,30`

PRL	**Purpose**	Specify a 2D query point relative to tile or display coordinate system.
	Syntax	PRL=*T,X,Y*
	Input Parameters	RANGE *T* The T tile number RANGE *X* The x coordinate RANGE *Y* The y coordinate
	Notes	**PRL** either specified a coordinate in a given tile, or if the tile number *T*=-1 then in the display coordinates
	Response	none
	Range	*T*=-1 .. maxtile *X,Y* are limited to the tile or section size
	Example	⇒PRL=2,20,1
	Default value	none
DPT	**Purpose**	Specify the rendering depth
	Syntax	DPT=*depth*
	Input Parameters	FLOAT *depth* rendering depth for projection rendering modes, depth zero implies no limit
	Response	none
	Example	⇒DPT=10
	Default value	0
RMD	**Purpose**	Specify the rendering mode
	Syntax	RMD=*mode*
	Input Parameters	*mode* ∈ SECT | PRJN | PRJD | PRJV The rendering mode
	Response	none
	Example	⇒RMD=PRJV
	Default value	SECT

ROL	**Purpose**	Specify the roll angle of the sectioning rotation
	Syntax	`ROL=`*`angle`*
	Input Parameters	`FLOAT` *`angle`* rotation angle in degrees
	Response	none
	Example	⇒`ROL=20.0`
	Range	`0.0-360.0`
	Default value	`0.0`
SCL	**Purpose**	Specify the scale used in the sectioning transformation.
	Syntax	`SCL=`*`scale`*
	Input Parameters	`FLOAT` *`scale`* the scale. A values over one is up, while lower than one is down-scaling.
	Response	none
	Example	⇒`SCL=2.5`
	Range	positive
	Default value	`1.0`
UPV	**Purpose**	Specify the up vector for the `UP_IS_UP` mode.
	Syntax	`UPV=`*`X,Y,Z`*
	Input Parameters	`FLOAT` *`X`* The x component `FLOAT` *`Y`* The y component `FLOAT` *`Z`* The z component
	Response	none
	Example	⇒`UPV=0,-1,2`
	Default value	`0.0,0.0,-1.0`
YAW	**Purpose**	Specify the yaw angle of the sectioning rotation.
	Syntax	`YAW=`*`angle`*
	Input Parameters	`FLOAT` *`angle`* rotation angle in degrees
	Response	none
	Example	⇒`YAW=2.0`
	Range	`0.0-360.0`
	Default value	`0.0`

PTL	**Purpose**	Retrieve a tile as a PNG image. This command is equivalent to the JTL command returning JPEG tiles.
	Syntax	`PTL=`*`res,tile`*
	Input Parameters	`INT` *`res`* resolution `INT` *`tile`* tile number
	Response	Requested tile *`tile`* on the resolution *`res`*
	Example	⇒`PTL=1,2`
	Default value	`0,0`
CVT	**Purpose**	Request an image to be returned as a composed image. CVT accepts JPG and PNG format requests.
	Syntax	`CVT=`*`format`*
	Input Parameters	`PNG|JPEG` *`format`* output format
	Response	Requested image
	Example	⇒`CVT=png`
	Default value	`jpeg`

SEL	**Purpose**	Specify a component of a compound object to be displayed and its colour. The component can be specified either by it's index or by a image processing expression which can combine the components through the use of image processing operators. Where a Woolz object is not a compound object, the SEL command will regard it as a compound object with a single component of index zero. Image processing expressions are explained in 2.4. If the command is not present then the first object is returned. Multiple **SEL** will stack up the section in the request order.
	Syntax	`SEL=`*`E`*`,`*`R`*`,`*`G`*`,`*`B`*`,`*`A`* `SEL=`*`E`*`,`*`R`*`,`*`G`*`,`*`B`* `SEL=`*`E`*`,`*`A`* `SEL=`*`E`*
	Input Parameters	`UINT` *`E`* compound object index or image processing expression `UINT` *`R`* red value `UINT` *`G`* green value `UINT` *`B`* blue value `UINT` *`A`* alpha value
	Response	none
	Example	⇒`SEL=0,255,0,255,127`
	Default value	`0,0,0,0,0`

2.3 Extended object reference

IIP-server	**Purpose**	Identify if WlzIIPSrv is running.
	Syntax	`IIP-server`
	Response	`IIP-server:255.552255`
	Example	⇒`OBJ=IIP-server` ⇐`IIP-server:255.552255`
	Note	This object should be used to verify if the IIP server has Woolz sectioning capabilities

Max-size	**Purpose**	Return the size of the section. For a Woolz object, the size is dependent on the sectioning parameters
	Syntax	`Max-size`
	Response	`Max-size:`*`width height`* `INT` *`width`* The width in pixels of the section at the current scale `INT` *`height`* The height in pixels of the section at the current scale
	Example	⇒`OBJ=Max-size` ⇐`Max-size:512 1024`
	Note	The size is dependent on the viewing plane defining parameters
Tile-size	**Purpose**	Return the size of a tile.
	Syntax	`Tile-size`
	Response	`Tile-size:`*`width height`* `INT` *`width`* The width in pixels of the tile `INT` *`height`* The height in pixels of the tile
	Example	⇒`OBJ=Tile-size` ⇐`Tile-size:64 64`
	Note	The size is constant throughout the life of the server (see also section 6.4).

Wlz-true-voxel-size	**Purpose**	Returns the voxel size of the object.
	Syntax	`Wlz-true-voxel-size`
	Response	`Wlz-true-voxel-size:`*`X Y Z`* `FLOAT` *`X`* The x size `FLOAT` *`Y`* The y size `FLOAT` *`Z`* The z size
	Example	⇒`OBJ=Wlz-true-voxel-size` ⇐`Wlz-true-voxel-size:2 1 2.2`
	Note	The voxel size is object specific, but view independent.
Wlz-volume	**Purpose**	Returns the volume of the object.
	Syntax	`Wlz-volume`
	Response	`Wlz-volume:`*`volume`* `INT` *`volume`* The volume of the Woolz object
	Example	⇒`OBJ=Wlz-volume` ⇐`Wlz-volume:748`
	Note	The volume is object specific, but view independent, with the volume being of the object or first selection if it exists.

Wlz-n-components	**Purpose**	Returns the number of components of a compound object.
	Syntax	`Wlz-n-components`
	Response	`Wlz-n-components:`*`n`* `INT` *`n`* The number of components of the Woolz object
	Example	⇒`OBJ=Wlz-n-components` ⇐`Wlz-n-components:3`
	Note	The number of components will be 1 if the object is not a compound object or is compound and only has a single component; in other cases the number of components will be greater for a valid object.
Wlz-distance-range	**Purpose**	Returns the range of the sectioning plane distance.
	Syntax	`Wlz-distance-range`
	Response	`Wlz-distance-range:`*`min max`* `FLOAT` *`min`* The minimum distance `FLOAT` *`max`* The maximum distance
	Example	⇒`OBJ=Wlz-distance-range` ⇐`Wlz-distance-range:-20 80`
	Note	The distance range is view-dependent.
Wlz-sectioning-angles	**Purpose**	Returns in degrees pitch, yaw and roll angles of the rotation of the sectioning plane.
	Syntax	`Wlz-sectioning-angles`
	Response	`Wlz-sectioning-angles:`*`pitch yaw roll`* `FLOAT` *`pitch`* The pitch angle of viewing plane rotation `FLOAT` *`yaw`* The yaw angle of viewing plane rotation `FLOAT` *`roll`* The roll angle of viewing plane rotation
	Example	⇒`OBJ=Wlz-sectioning-angles` ⇐`Wlz-sectioning-angles:0 40 30`

Wlz-3d-bounding-box	**Purpose**	Returns The first and last plane, line and column number of the object.
	Syntax	`Wlz-3d-bounding-box`
	Response	`Wlz-3d-bounding-box:`*`plane1 lastpl line1 lastln colum1 lastcl`* `FLOAT` *`plane1`* The first plane number of the object `FLOAT` *`lastpl`* The last plane number of the object `FLOAT` *`line1`* The first line number of the object `FLOAT` *`lastln`* The last line number of the object `FLOAT` *`colum1`* The first column number of the object `FLOAT` *`lastcl`* The last column number of the object
	Example	⇒`OBJ=Wlz-3d-bounding-box` ⇐`Wlz-3d-bounding-box:0 10 -15 15 30 90`

Wlz-transformed-3d-bounding-box	**Purpose**	Returns The first and last plane, line and column number of the object after a section transform.
	Syntax	`Wlz-transformed-3d-bounding-box`
	Response	`Wlz-transformed-3d-bounding-box:`*`plane1 lastpl line1 lastln colum1 lastcl`* `FLOAT` *`plane1`* The first section plane number `FLOAT` *`lastpl`* The last section plane number `FLOAT` *`line1`* The first line number of the section `FLOAT` *`lastln`* The last line number of the section `FLOAT` *`colum1`* The first column number of the section `FLOAT` *`lastcl`* The last column number of the section
	Example	⇒`OBJ=Wlz-transformed-3d-bounding-box` ⇐`Wlz-transformed-3d-bounding-box:10 100 -23 12 308 490`
Wlz-coordinate-3D	**Purpose**	Returns the 3D object coordinates defined in 2D by the **PRL** command.
	Syntax	`Wlz-coordinate-3D`
	Response	`Wlz-coordinate-3D:`*`X Y Z`* `FLOAT` *`X`* The x coordinate `FLOAT` *`Y`* The y coordinate `FLOAT` *`Z`* The z coordinate
	Example	⇒`OBJ=Wlz-coordinate-3D` ⇐`Wlz-coordinate-3D:20 30 10`

Wlz-grey-stats	**Purpose**	Computes and returns simple statistics of the grey values of a Woolz object.
	Syntax	`Wlz-grey-stats`
	Response	`Wlz-grey-stats:`*`n t gl gu sum ss mean stddev`* or `Wlz-grey-stats:`*`NULL`* `UINT` *`n`* Number of image values in the Woolz object. `WlzGreyType` *`t`* The Woolz image value type, with t ∈ *`WLZ_GREY_INT`* | *`WLZ_GREY_SHORT`* | *`WLZ_GREY_UBYTE`* | *`WLZ_GREY_FLOAT`* | *`WLZ_GREY_DOUBLE`* | *`WLZ_GREY_RGBA`* `FLOAT` *`gl`* Minimum image value. `FLOAT` *`gu`* Maximum image value. `FLOAT` *`sum`* Sum of image values. `FLOAT` *`ss`* Sum of squares of image values. `FLOAT` *`mean`* Mean image value. `FLOAT` *`stddev`* Standard deviation of image values.
	Example	⇒`OBJ=Wlz-grey-value` ⇐`Wlz-grey-stats: 489600 UBYTE 0 255 8.5857e+06 1.42684e+09 17.5362 51.0567`
	Note	For RGBα Woolz objects the computed values will be for the modulus of the image values. The statistics are of the object or first selection if it exists. When the object of first selection does not have image values *`NULL`* will be returned.

Wlz-grey-value	**Purpose**	Returns in grey or RGB value of a point specified either the **PRL** or the **PAB** commands.
	Syntax	`Wlz-grey-value`
	Response	`Wlz-grey-value:`*`grey`* or `Wlz-grey-value:`*`R G B`* `INT` *`grey`* The grey value of the pixel `INT` *`R`* The red channel of the pixel `INT` *`G`* The green channel of the pixel `INT` *`B`* The blue channel of the pixel
	Example	⇒`OBJ=Wlz-grey-value` ⇐`Wlz-grey-value:0 255 10`
	Note	The returned values are in the range of 0 to 255.
Wlz-transformed-coordinate-3d	**Purpose**	Returns the the display coordinates and displacement from the sectioning plane of a 3D point defined by **PAB**
	Syntax	`Wlz-transformed-coordinate-3d`
	Response	`Wlz-transformed-coordinate-3d:`*`X Y D`* `INT` *`X`* The x coordinate in display frame `INT` *`Y`* The y coordinate in display frame `FLOAT` *`D`* The the signed distance from the sectioning plane
	Example	⇒`OBJ=Wlz-transformed-coordinate-3d` ⇐`Wlz-transformed-coordinate-3d:30 40 60.0002`

Wlz-foreground-objects	**Purpose**	The components of the compound object that 2D/3D query point is a foreground
	Syntax	`Wlz-foreground-objects`
	Response	`Wlz-transformed-coordinate-3d:`O_1 O_2 ... O_n `INT` O_1 The compound object index that a point is a foreground pixel
	Example	⇒`OBJ=Wlz-foreground-objects` ⇐`Wlz-foreground-objects: 0 2 5 10`

Object queries **Author**, **Copyright**, **Create-dtm**, **Subject** and **App-name** return *author/ copyright/ create-dtm/ subject/ app-name unknown* since they are not defined for a Woolz object.

2.4 Image Processing Expressions

Image processing expressions may be built using the components of Woolz compound objects and basic image processing operators. The operators supported are: background, difference, dilation, domain, erosion, fill, intersection, occupancy, setvalue, threshold, transfer and union. Image processing expressions are built from compound object indices and the image processing operators. Two of the image processing operators (occupancy and union) may given a list indices in their expressions. The occupancy operator may also be given no indices which implies all components of the compound object. Whether complex selections that use image processing operations are allowed can be controlled using the COMPLEX_SELECTION environment variable. The image processing expressions can be written using the syntax defined in table 3. This allows expressions such as:

`erosion(1,2)` erodes the domain of the object with index one in the compound object by two voxels.

`diff(dilation(1,2),3)` dilates the domain of the object with index one in the compound object by two voxels and then computes the difference between that and the object with index 3.

`domain(threshold(0,200,ge))` creates a domain from the object with index 0 where all voxels have values greater than or equal to 200. Without the domain operator threshold would give a object with image values.

exp list	:=	(*exp*|*idx list*)(,*exp list*)
exp	:=	*idx*|
		background(*exp*,*val*) |
		diff(*exp*,*exp*) |
		dilation(*exp*,*uint*) |
		domain(*exp*) |
		erosion(*exp*,*uint*) |
		fill(*exp*) |
		intersect(*exp list*,*exp list*) |
		occupancy(*index list*?)|
		setvalue(*exp*,*val*) |
		threshold(*exp*,*val*,*cmp*) |
		transfer(*exp*,*exp*) |
		union(*exp list* |*idx list*,*exp list* |*idx list*)
idx list	:=	(*uint* |*idx range* |*idx list*,*idx list*)
idx range	:=	*uint*-*uint*
idx	:=	*uint*
val	:=	(*int*)|(*float*)([eE]*int*)?
real	:=	*int*[.]*uint*
int	:=	-?*uint*
uint	:=	[0-9]+
cmp	:=	(lt)|(le)|(eq)|(ge)|(gt)

Table 3: Syntax for image processing expressions.

The image processing operators available and their parameters are described in table 4.

3 HTML query examples

Tile request

```
http://localhost/fcgi-bin/iipsrv.fcgi?
  WLZ=/home/bill/pics/lobster3d.wlz&
  QLT=50&JTL=1,0
```

Returns a jpeg tile 0 with a quality factor of 50%. The section default parameters, mode UP_IS_UP and zero distance and viewing angles are used.

Sectioning plane distance

Operator	**Complexity**	**Description**
background(*exp*,*value*)	1	Sets the background value (value outside the expresion object's domain) to the given value.
diff(*exp*,*exp*)	2	The difference between the two given domains.
dilation(*exp*,*radius*)	2	The dilation of the domain by *radius* voxels.
domain(*exp*)	1	The domain of the expresion's object.
erosion(*exp*,*radius*)	2	The erosion of the domain by *radius* voxels.
fill(*exp*)	2	Fills all holes not connected to the outside in the given expression's domain.
intersect(*exp list*)	2	The intersection of the domains in the given lists.
occupancy(*idx list*?)	2	The domain occupancy in the optional list.
setvalue(*exp*,*value*)	2	Creates an object with the domain of the given expression and all values set to the given value.
threshold(*exp*,*value*, *comparison*)	2	Creates an object where the image values satisfy the given *value* and *comparison*. Here the value is floating point and valid comparisons are `lt` (less than), `le` (less than or equal), `eq` (equal), `ge` (greater than or equal) and `gt)` (greater than).
transfer(*exp*,*exp*)	2	Creates a new object with the domain of the first expression's object and the values of the second. Values outside the domain of the second but within the first are set to the second's background value.
union(*exp list*|*idx list*, *exp list*|*idx list*)	2	The union of the domains in the given lists.

Table 4: Descriptions of image processing operators with complexity values

.

```
http://localhost/fcgi-bin/iipsrv.fcgi?
  WLZ=/home/bill/pics/lobster3d.wlz&DST=8&QLT=50&PTL=1,0
```

As above, but the sectioning plane distance is 8 and the returned tile has been compressed with png not jpeg format.

Sectioning mode, plane distance and angle

```
http://localhost/fcgi-bin/iipsrv.fcgi?DST=40&YAW=61.5&PIT=3&ROL=0&MOD=ZETA&
  WLZ=/home/zsolth/small.wlz&QLT=50&CVT=jpeg
```

Returns the whole section with the distance 40, yaw angle 61.5, pitch 3, roll 0 degrees, for a plane at distance 40

Distance range query

```
http://localhost/fcgi-bin/iipsrv.fcgi?YAW=61&PIT=3&ROL=0&MOD=ZETA&
  WLZ=/home/zsolth/small.wlz&OBJ=Wlz-distance-range
```

results in

```
Wlz-distance-range:0 171
```

3D object coordinate query

```
http://localhost//fcgi-bin/l.fcgi?WLZ=/home/zsolth/fif/wlz/ts14.wlz&
  DST=200&YAW=10&PIT=3&PRL=0,30,40&OBJ=Wlz-coordinate-3d
```

results in

```
Wlz-coordinate-3d:418.43 53.5708 15.9916
```

The (30,40) display coordinate in sectioning plane DST=200 has 3D object coordinates (418.43,53.5708,15.9916).

3D coordinate projection to a section

```
http://localhost//fcgi-bin/l.fcgi?WLZ=/home/zsolth/fif/wlz/ts14.wlz&
  DST=130&YAW=10&PIT=3&PAB=418.43,53.5708,15.9916&OBJ=Wlz-tranformed-coordinate-
```

results in

```
Wlz-transformed-coordinate-3d:30 40 70.0002
```

The (418.43,53.5708,15.9916) object coordinate in sectioning plane DST=130 has display coordinates (30,40) and has a distance 70.0002 from the plane.

Sectioning, selection and image processing operations

```
http://localhost/fcgi-bin/iipsrv.fcgi?DST=40&YAW=61.5&PIT=3&ROL=0&MOD=ZETA&
WLZ=/home/bill/compound.wlz&
SEL=(dilation(union(1-3)),255,0,0,255)&
QLT=50&CVT=jpeg
```

The section is selected as described above and a domain is created that is the dilated union of components 1, 2 and 3 of the compound object.

4 WlzIIPSrv coding

4.1 Architecture

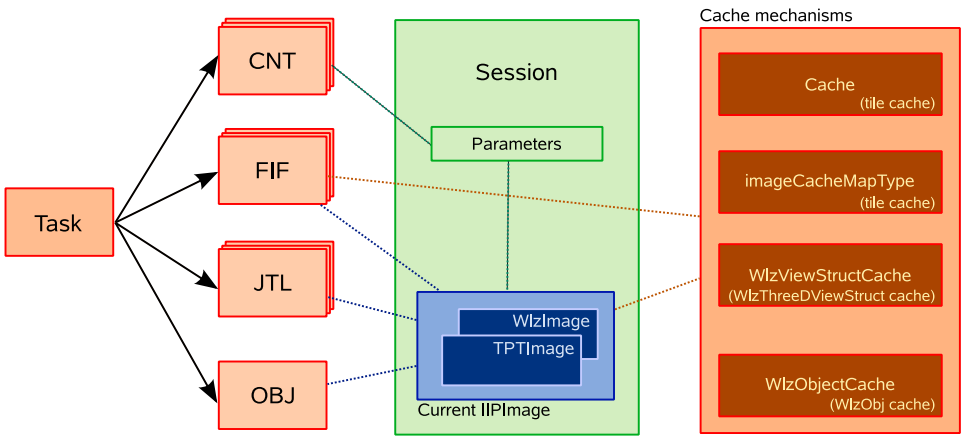

Figure 2: wlziipsrv version 0.9.7 architecture

Figure 2 depicts the architecture of the WlzIIPSrv this being inherited from the original IIPSrv and explaining both. Blocks in the figure represent either C++ classes or structures.

The **Task** class is the base of all command classes that serve independently each IIP command. Its static `factory(string type)` method creates an instance of the appropriate class that serves the `type` IIP command.

Figure 2 shows some of the classes implementing the IIP commands, all derived form the `Task`. Four command types can be identified:

Parameter setting commands such **CNT** and **ROL** pass a parameter to the server. The parameter value is stored by the command in the `Session` structure.

Object defining commands such **FIF** and **WLZ** receive the file name of an object and creates the necessary structures to handle an object. In WlzIIPSrv, either `TPTImage` and `WlzImage` classes store Tiled Pyramidal TIFFs and Woolz objects.

Image providers such **JTL**, **PTL** and **CVT** generate a tile or a full image corresponding to the previously set parameters, stored in the **Session** structure.

Parameter enquiry with **OBJ** command returns an parameter computed or previously specified.

The `Session` structure stores all session related parameters, including the current image object.

To reduce the computation overhead, three types of caches are used:

- for tiles (`RawTile`),
- for flat images (`IIPImage`)
- for Woolz objects and view structures (`WlzObj`).

The interactions between classes, figure 2, are multiple:

- task creates the requested command class,
- commands set session related parameters and the current object
- the **OBJ** command provides the current object parameters,
- before loading from disc, when an operation requires access object data the appropriate cache structure is checked first. Also, when a tile is requested, if it parameters result in a cache hit then the cached tile is returned. The a Woolz view structures are cached and looked before a section is generated.

4.2 IIPSrv and WlzIIPSrv beside the IIP specification

IIP [4, p.25] defines a set of mandatory and optional commands that an IIP server should implements. The IIPSrv implements a subset of this commands, while also adds extra commands. WlzIIPSrv further extends the command set, while some of the original IIP commands are incompatible with Woolz objects.

The list of commands from Table 5 compares command sets of the IIP specification, the IIPSrv and the WlzIIPSrv with supported (S), unsupported (N), partially implemented (P) and commands in an unknown, an undocumented or nonfunctional state (U).

5 Woolz IIP Proxy

The Woolz IIP Proxy filters and forwards IIP requests to one or more WlzIIP servers. The requests conform the FCGI protocol. Though it was designated to work for IIP and Woolz requests, it is generic and can be applied to any FCGI request. Therefore, it is possible to chain multiple proxies.

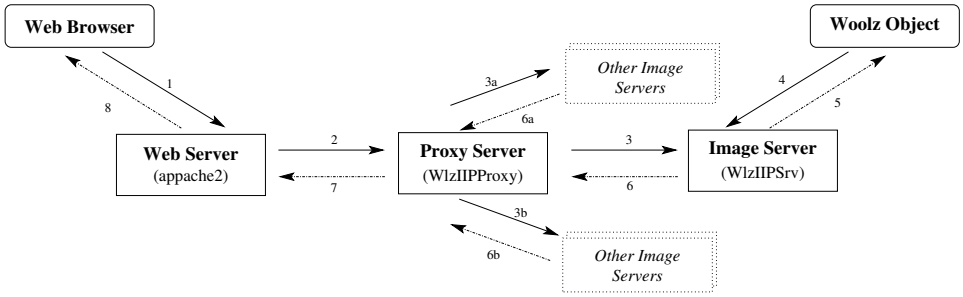

Figure 3: Architecture of Woolz IIP Server using a proxy server. The proxy forwards the user web requests served first by apache to the individual IIP Servers that have direct access to the Woolz Object. The numbered lines show the ordering of the requests (continuous line) and the replies (dotted lines).

The WlzIIPProxy is an independent program running on the proxy server. Apache2 server forwards the FCGI request to this sever, on a configurable port. WlzIIPSrv check the html request string (the FCGI_PARAMS packet QUERY_STRING parameter) and if any remote FCGI server definition string is a substring of the request then request is forwarded to this server. If there is no hit then the request is forwarded to the first FCGI remote server. The default setup of the WlzIIPProxy architecture is shown in figure3

5.1 WlzIIPProxy options

Usage:

```
WlzIIPProxy [-p<portnumber>] [-c<conf_filename>] [-l<log_filename>]
```

Command	IIP spec	IIPSrv	WlzIIPSrv
AFN	S	N	N
CIN	S	N	N
CNT	S	S	S
CTW	S	N	N
CVT	S	S	S
FIF	S	S	S
FTR	S	N	N
HEI	S	U	U
ICC	S	S	U
JTL	S	S	P
OBJ	S	S	S
QLT	S	S	S
RAR	S	N	N
RFM	S	N	N
RGN	S	S	S
ROI	S	S	S
RST	S	N	N
SDS	S	S	N
TIL	S	S	P
WID	S	U	U
JTLS	N	S	N
SHD	N	S	U
DST	N	N	S
FPL	N	N	S
FXP	N	N	S
FXT	N	N	S
MAP	N	N	S
MOD	N	N	S
PAB	N	N	S
PIT	N	N	S
PRL	N	N	S
PTL	N	N	S
DPT	N	N	S
RMD	N	N	S
ROL	N	N	S
SCL	N	N	S
SEL	N	N	S
UPV	N	N	S
WLZ	N	N	S
YAW	N	N	S
Affine-transform	S	N	N
App-name	S	U	P
Aspect-ratio	S	N	N
Author	S	U	N
Basic-info	S	S	S
Colorspace	S	S	U
Color-twist	S	N	N
Comment	S	U	N
Comp-group	S	N	N
Contrast-adjust	S	N	N
Copyright	S	U	S
Create-dtm	S	U	N
Edit-time	S	U	N
File-class-id	S	N	N
Filtering-value	S	N	N
ICC-profile	S	N	N
IIP-opt-comm	S	S	S
IIP-opt-obj	S	S	S
IIP-server	S	S	S
IIP-socket	S	N	N
IIP	S	P	P
Keywords	S	U	N
Last-author	S	U	N
Last-printed	S	U	N
Last-save-dtm	S	U	N
Max-size	S	S	S
Property	S	N	N
Render-path	S	N	N
Resolution-number	S	S	S
Rev-number	S	U	N
ROI	S	N	N
Security	S	N	N
Stream	S	N	N
Subject	S	U	P
Summary-info	S	U	P
Title	S	U	N
View-info	S	N	N
Horizontal-views	N	S	U
Tile-size	N	S	S
Vertical-views	N	S	U
Wlz-3d-bounding-box	N	N	S
Wlz-coordinate-3D	N	N	S
Wlz-distance-range	N	N	S
Wlz-foreground-objects	N	N	S
Wlz-grey-stats	N	N	S
Wlz-grey-value	N	N	S
Wlz-n-components	N	N	S
Wlz-sectioning-angles	N	N	S
Wlz-transformed-3d-bounding-box	N	N	S
Wlz-transformed-coordinate-3d	N	N	S
Wlz-true-voxel-size	N	N	S
Wlz-volume	N	N	S

```
[-v<loglevel>] [-h]
```

where the options are:

- -p Port number. Default value: 123777
- -c Configuration file containing WLZ name to server name mapping. Default value: WlzIIPProxy.conf
- -l Log file name. -v with a value greater than 0 must be used. Default value: /tmp/WlzIIPProxy.log
- -v Log level. Default value: 0
- -h Help, prints usage message.

The log level is 0 to 3 with

- 0 – no log,
- 1 – system startup, shutdown, error messages,
- 2 – as fcgi connection messages,
- 3 – as level 2 and received/sent packet types.

The configuration file has three columns for:

1. search string,
2. server name (or ip),
3. port number.

6 WlzIIPSrv installation

6.1 Install FastCGI

Download and install FastCGI [LIB]. WlzIIPSrv was tested with fcgi-2.4.0 only.

6.2 Source code

Obtain the source files from github:

`https://github.com/ma-tech/WlzIIPSrv`

The following packages are also required:

`https://github.com/ma-tech/External`
`https://github.com/ma-tech/Woolz`

6.3 Compiling

Having already built the required external libraries including the Woolz libraries; within the `WlzIIPSrv` root directory edit and execute the build script:

```
cp build.sh mybuild.sh
vi mybuild.sh
./mybuild.sh
```

The build script can be edited, but most often this will only need to be done to change the install prefix. The Woolz libraries should be build with external file format support enabled.

For doxygen documentation run `make doc`. The documentation is generated into the `Docs` directory.

Installing implies moving the `src/wlziipsrv.fcgi` into the server's fcgi directory:

```
cp src/wlziipsrv.fcgi /opt/apache/fcgi-bin/
```

Do not forget to set read and execute access modes!

6.4 Customisable parameters

In addition to the IIPSrv [Pillay and Pitzalis, p.25] configuration parameters, `WlzIIPSrv` allows with the parameters from Table 6 changing the logging, caches and tile sizes.

The example configuration from appendix A defines a view structure cache with 200 structures, a 1500MB Woolz object cache and 100×100 tile size.

7 WlzIIPProxy installation

- Download and compile / install FastCGI [LIB].
- Get WlzIIPProxy source code from the CVS repository:

  ```
  cvs checkout -P src/Applications/WlzIIPProxy
  ```

- Configure and compile WlzIIPProxy: Run from the `WlzIIPSrv` root directory

  ```
  aclocal; autoheader; automake; autoconf; ./configure; make
  ```

Parameter	Description	Default value
LOGFILE	Location of log file.	/tmp/iipsrv.log
LOGLEVEL	Log level (EMERG, FATAL, ALERT, CRIT, ERROR, WARN, NOTICE, INFO, DEBUG).	WARN
FILESYSTEM_PREFIX	Prefix directory for images.	”
MAX_WLZOBJ_CACHE_SIZE	Maximum Woolz object cache size in MBs.	1024
MAX_WLZOBJ_CACHE_COUNT	Maximum number of Woolz object in cache.	100
WLZ_TILE_WIDTH	Tile width in pixels.	100
WLZ_TILE_HEIGHT	Tile height in pixels.	100
COMPLEX_SELECTION	Controls complex selections with values: 0 - No complex selections, 1 - All low cost selections, 2 - All selections.	0

Table 6: WlzIIPSrv extra configuration parameters

- Run WlzIIPProxy on the desired port and using the configuration file.

```
WlzIIPProxy -p <portnumber> -c <config_file>
```

- Configure apache2 on the web server:

  1. install mod_fastcgi (you might need to reinstall apache2) **Note:** the mod_fcgi module, provided in SUSE 10.3 can not make remote FCGI request therefore is not compatible with WlzIIPSrv. mod_fastcgi must be used.
  2. add to httpd.conf (/opt/apache/conf//httpd.conf):

```
FastCgiExternalServer <virtual_path_to_fcgi> -host <hostname>:<portnumber>
```

8 Acknowledgement

We gratefully acknowledge the support from NIH under grant #1R01MH070370-01A2.

A fcgi configuration

An example of the apache2 configuration (`/opt/apache/conf/httpd.conf`) is[1]

```
# Create a directory for the iipsrv binary
ScriptAlias /fcgi-bin/ "/opt/apache/fcgi-bin/"

# Set the options on that directory
<Directory "/opt/apache/fcgi-bin">
   AllowOverride None
   Options None
   Order allow,deny
   Allow from all
   # Set the module handler
   AddHandler fcgid-script .fcgi
</Directory>

# Initialise the FCGI server and set some default values
FastCgiServer /opt/apache/fcgi-bin/wlziipsrv.fcgi \
-initial-env LOGFILE=/opt/apache/logs/wlziipsrv.log \
-initial-env LOGLEVEL=WARN \
-initial-env JPEG_QUALITY=50 \
-initial-env FILESYSTEM_PREFIX=/export/images \
-initial-env MAX_CVT=3000 \
-initial-env MAX_WLZOBJ_CACHE_COUNT=100 \
-initial-env MAX_WLZOBJ_CACHE_SIZE=4000 \
-initial-env WLZ_TILE_WIDTH=100 \
-initial-env WLZ_TILE_HEIGHT=100 \
-initial-env COMPLEX_SELECTION=0
```

References

[EMA] The e-mouse atlas project.

[LIB] Fastcgi api (libfcgi).

[WOO] Woolz image processing system.

[1] This configuration file is compatible with mod_fcgi. For mod_fastcgi alternative format must be used (see IIPSrv web-page).

[4] I3A (1997). Internet imaging protocol.

[Pillay and Pitzalis] Pillay, R. and Pitzalis, D. IIPSrv.