

# Portfolio

## Finding Relevant Functions and their Usages

**Collin McMillan**<sup>1</sup>

Mark Grechanik<sup>2</sup>

Denys Poshyvanyk<sup>1</sup>

Qing Xie<sup>2</sup>

Chen Fu<sup>2</sup>



<sup>1</sup>College of William & Mary

<sup>2</sup>Accenture Technology Labs

# Virtues of Programmers:

- **Laziness**
- Impatience
- Hubris

Larry Wall,  
Inventor of Perl

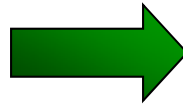


# Example Programming Task

**Write a utility for dithering mip map images that are used for rendering texture.**



Mip Maps



Dithering

# What Programmers Do: Use A Search Engine!

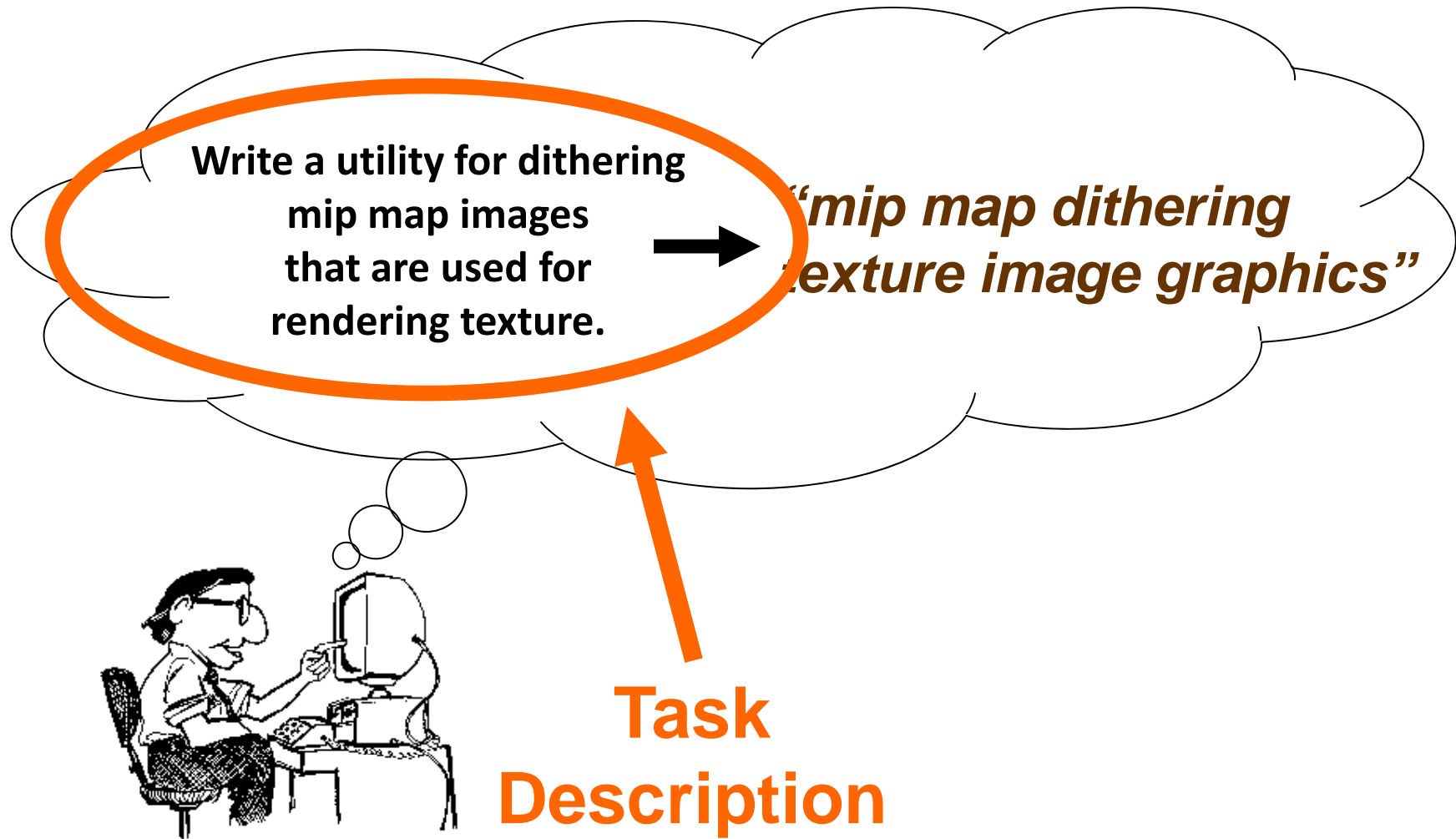
Write a utility for dithering  
mip map images  
that are used for  
rendering texture.



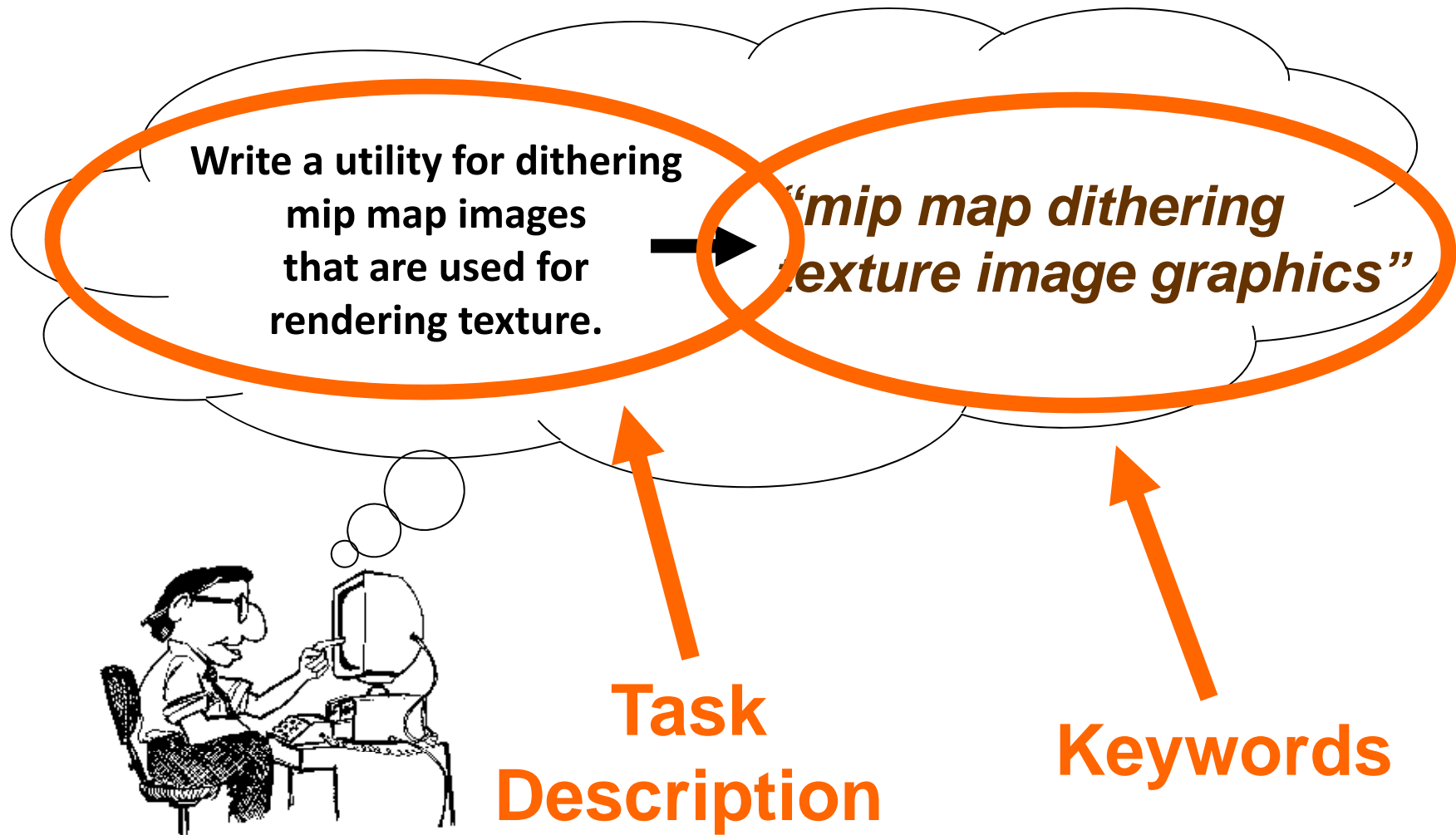
***“mip map dithering  
texture image graphics”***



# What Programmers Do: Use A Search Engine!



# What Programmers Do: Use A Search Engine!



# Example Source Code Search Engine

“mip map dithering  
texture image  
graphics”



```
1 | 2 | 3 | 4 | 5 | 6 |
mooeditprefs.h
/*
 * mooeditprefs.h
 *
 * Copyright (C) 2004-2007 by Yevgen Muntyan <muntyan@math.tamu.edu>

Language: C
(C) 2004-2007 by Yevgen Muntyan
LOC: 52
Spider_20090401_inc : mooedit (project search): .../medit-0.9.3.tar.bz2/medit-0.9.3/moo/mooedit/mooeditprefs.h

pngconf.h
/* pngconf.h - machine configurable file for libpng
 *
 * libpng 1.0.8 - July 24, 2000

Language: C
(c) 1995, 1996 Guy Eric Schalnat, Group 42, Inc.)...
LOC: 1021
Spider_291 : ApacheToolbox (project search): .../src/pdflib-4.0.2.tar.gz---/pdflib-4.0.2/png/pngconf.h

file.c
/*
Copyright (C) 1998, 1999, 2000, 2001, 2002, 2003, 2004, 2005, 2006 Matthew
```

# Example Source Code Search Engine

Query

“mip map dithering  
texture image  
graphics”



1 | 2 | 3 | 4 | 5 | 6 |

**mooeditprefs.h**

```
/*  
 * mooeditprefs.h  
 *  
 * Copyright (C) 2004-2007 by Yevgen Muntyan <muntyan@math.tamu.edu>  
 */
```

Language: C  
(C) 2004-2007 by Yevgen Muntyan  
LOC: 52  
Spider\_20090401\_inc : mooedit (project search): .../medit-0.9.3.tar.bz2/medit-0.9.3/moo/mooedit/mooeditprefs.h

**pngconf.h**

```
/* pngconf.h - machine configurable file for libpng  
 *  
 * libpng 1.0.8 - July 24, 2000  
 */
```

Language: C  
(c) 1995, 1996 Guy Eric Schalnat, Group 42, Inc.)...  
LOC: 1021  
Spider\_291 : ApacheToolbox (project search): .../src/pdflib-4.0.2.tar.gz---/pdflib-4.0.2/png/pngconf.h

**file.c**

```
/*  
 Copyright (C) 1998, 1999, 2000, 2001, 2002, 2003, 2004, 2005, 2006 Matthew
```



# Example Source Code Search Engine

Query

“mip map dithering  
texture image  
graphics”



List of Results

**mooeditprefs.h**

```
/*  
 * mooeditprefs.h  
 * Copyright (C) 2004-2007 by Yevgen Muntyan <muntyan@math.tamu.edu>
```

```
Language: C  
(C) 2004-2007 by Yevgen Muntyan  
LOC: 52  
Spider_20090401_inc : mooedit (project search): .../medit-0.9.3.tar.bz2/medit-0.9.3/moo/mooedit/mooedit
```

**pngconf.h**

```
/* pngconf.h - machine configurable file for libpng  
 *  
 * libpng 1.0.8 - July 24, 2001
```

```
Language: C  
(c) 1995, 1996 Guy Eric Schalnat, Group, Inc.)...  
LOC: 1021  
Spider_291 : ApacheToolbox (project search): .../src/pnglib-4.0.2.tar.gz---/pnglib-4.0.2/png/pngconf.h
```

**file.c**

```
/*  
 Copyright (C) 1998, 1999, 2000, 2001, 2002, 2003, 2004, 2005, 2006 Matthew
```

# Users Prefer Web Search to Code Search

Of 35 Professional Programmers we surveyed:

- 12 did not search for code online
- 19 did not use code search engines
- 14 cited “irrelevant results” as the reason
- Preferred **Web** Search to **Code** Search!

# Users Prefer Web Search to Code Search

Of 35 Professional Programmers we surveyed:

- 12 did not search for code online
- 19 did not use code search engines
- 14 cited “irrelevant results” as the reason
- Preferred **Web** Search to **Code** Search!

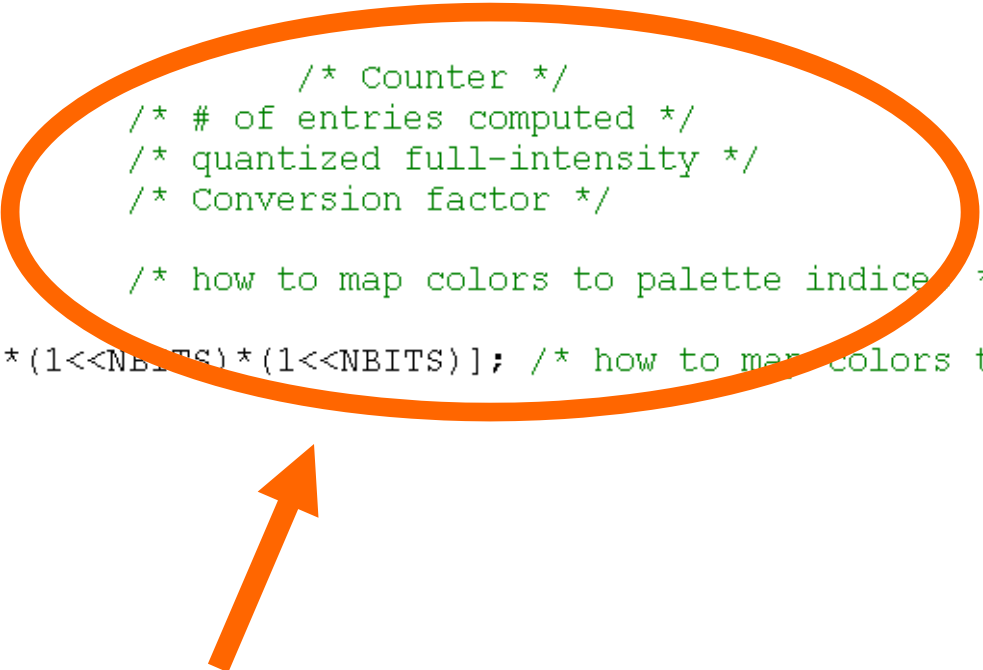
***How Could This Happen?***

# Search Result - A Relevant Function

```
111 /*
112  * Perform variance-based color quantization on a 24-bit image.
113  */
114 int
115 txMipPal256(TxMip *pxMip, TxMip *txMip, int format, FxU32 dither, FxU32 compression)
116 {
117     int        w, h;
118     int        i;                                /* Counter */
119     int        OutColors;                        /* # of entries computed */
120     int        Colormax;                        /* quantized full-intensity */
121     float      Cfactor;                        /* Conversion factor */
122 #if 0
123     uchar      *rgbmap;                        /* how to map colors to palette indices */
124 #else
125     static uchar rgbmap[(1<<NBITS)*(1<<NBITS)*(1<<NBITS)]; /* how to map colors to pa
126 #endif
127     int        pixsize;
```

# Search Result - A Relevant Function

```
111 /*
112  * Perform variance-based color quantization on a 24-bit image.
113  */
114 int
115 txMipPal256(TxMip *pxMip, TxMip *txMip, int format, FxU32 dither, FxU32 compression)
116 {
117     int        w, h;
118     int        i;
119     int        OutColors;
120     int        Colormax;
121     float      Cfactor;
122     #if 0
123     uchar      *rgbmap;
124     #else
125     static uchar rgbmap[(1<<NBITS)*(1<<NBITS)*(1<<NBITS)]; /* how to map colors to pa
126     #endif
127     int        pixsize;
```



/\* Counter \*/  
/\* # of entries computed \*/  
/\* quantized full-intensity \*/  
/\* Conversion factor \*/  
  
/\* how to map colors to palette indices \*/

**Keywords in Comments**

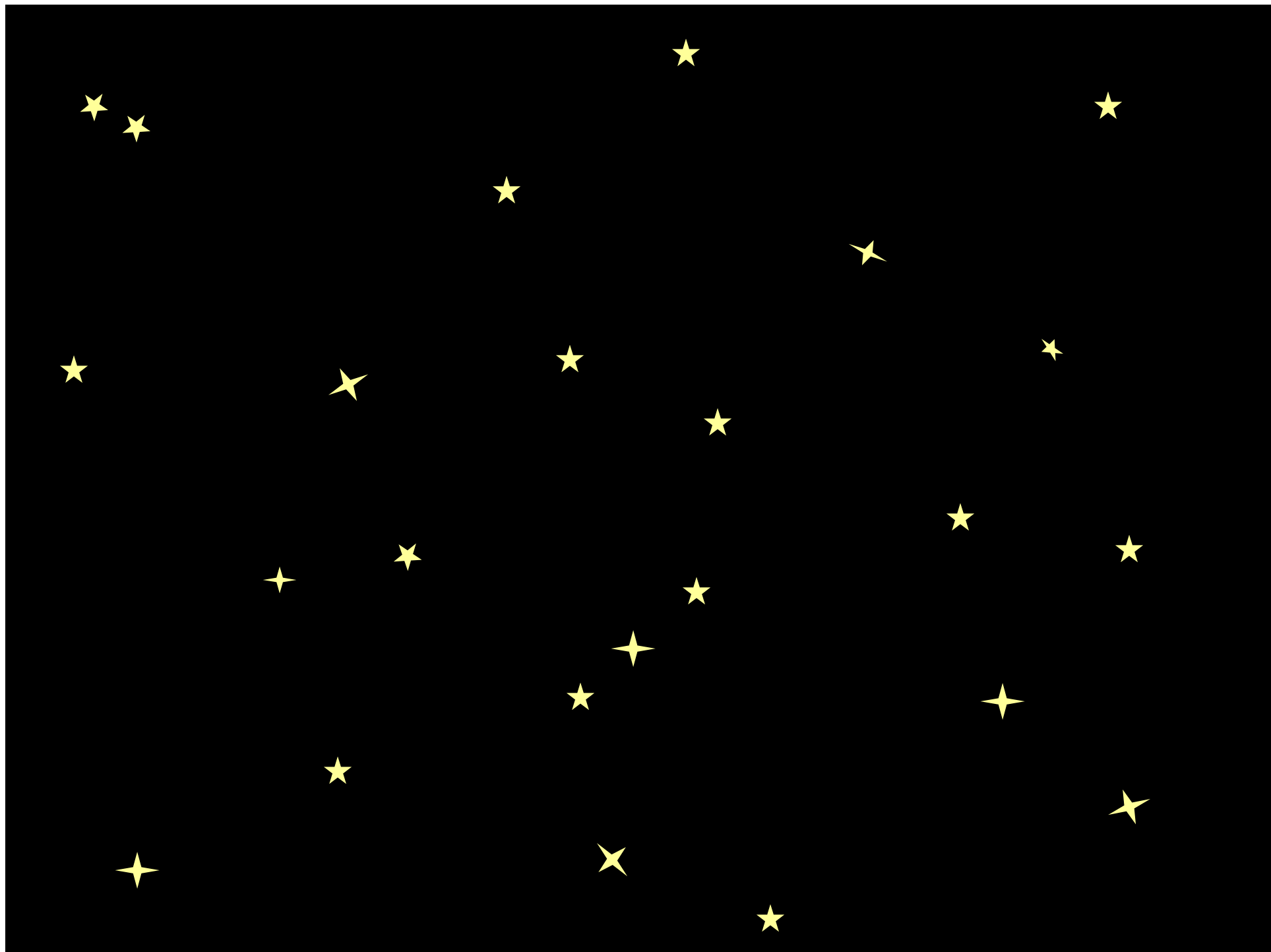
# Find Usages of Relevant Functions

```
392     * Get a 256 color palette, to be used as samples
393     * Incidentally, convert src 32 bit image to dst 8 bit indexed image,
394     * with indices referring to the 256 color palette.
395     * Also incidentally, pack the alpha channel if necessary.
396     */
397     if( txVerbose )
398     {
399         printf("NCC Neural nets..."); fflush(stdout);
400     }
401     pxMip->format = (format == GR_TEXFMT_YIQ_422) ? GR_TEXFMT_P_8 :
402         GR_TEXFMT_AP_88;
403     ncolors = txMipPal256(pxMip, txMip, pxMip->format, 0, 0);
404     if( txVerbose )
405     {
406         printf("%d samples...", ncolors); fflush(stdout);
407     }
408     txMapPal256toYAB((FxU32 *)yabTable, (FxU8 *)map, ncolors, (FxU32 *)pxMip->pal);
409     if( txVerbose )
410     {
411         printf("eMax=(%3ld%3ld%3ld)...eAvg=(%3ld%3ld%3ld)\n",
412             errG, errR, errB,
413             totG/ncolors, totR/ncolors, totB/ncolors
```

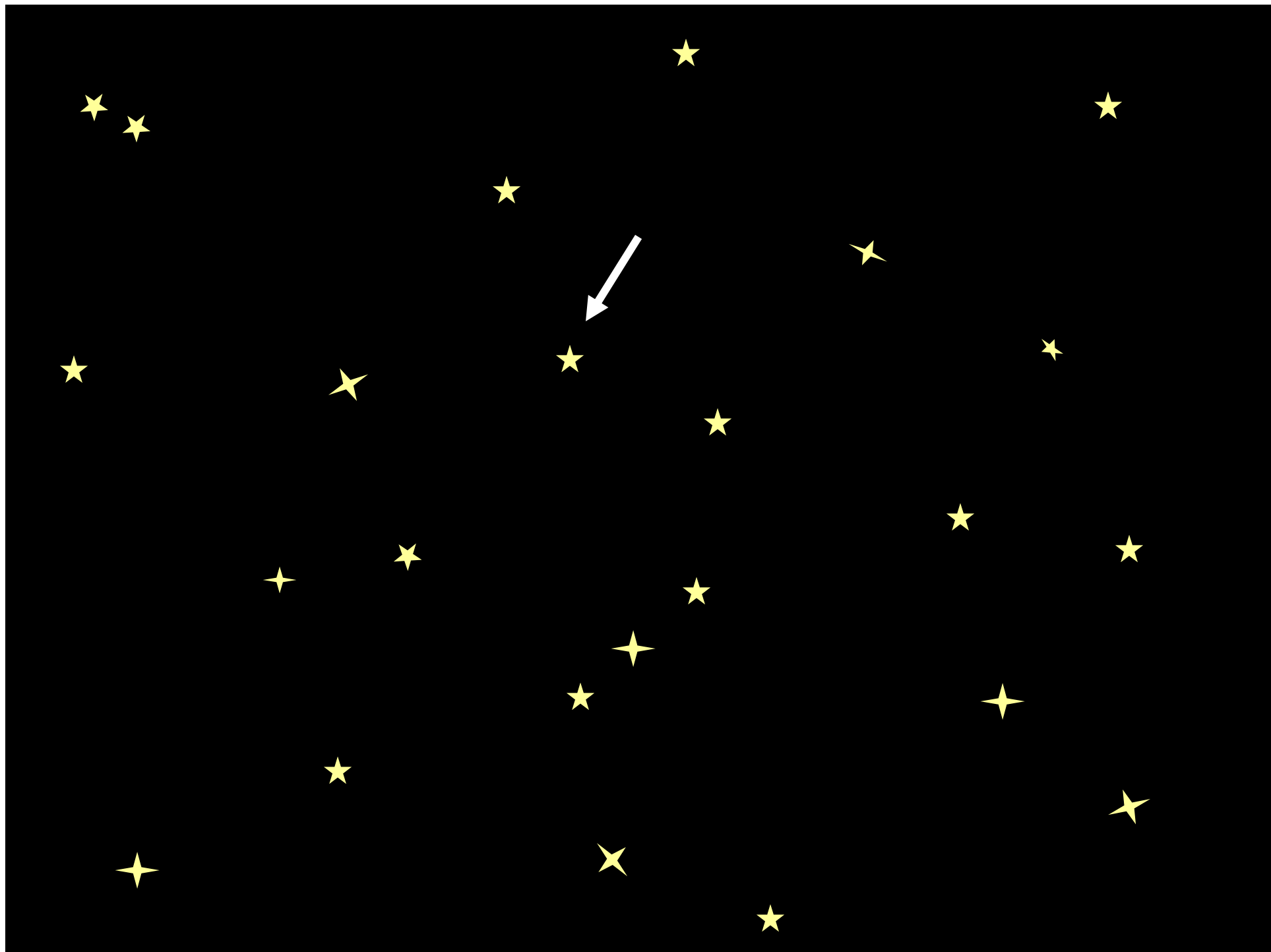
# Find Usages of Relevant Functions

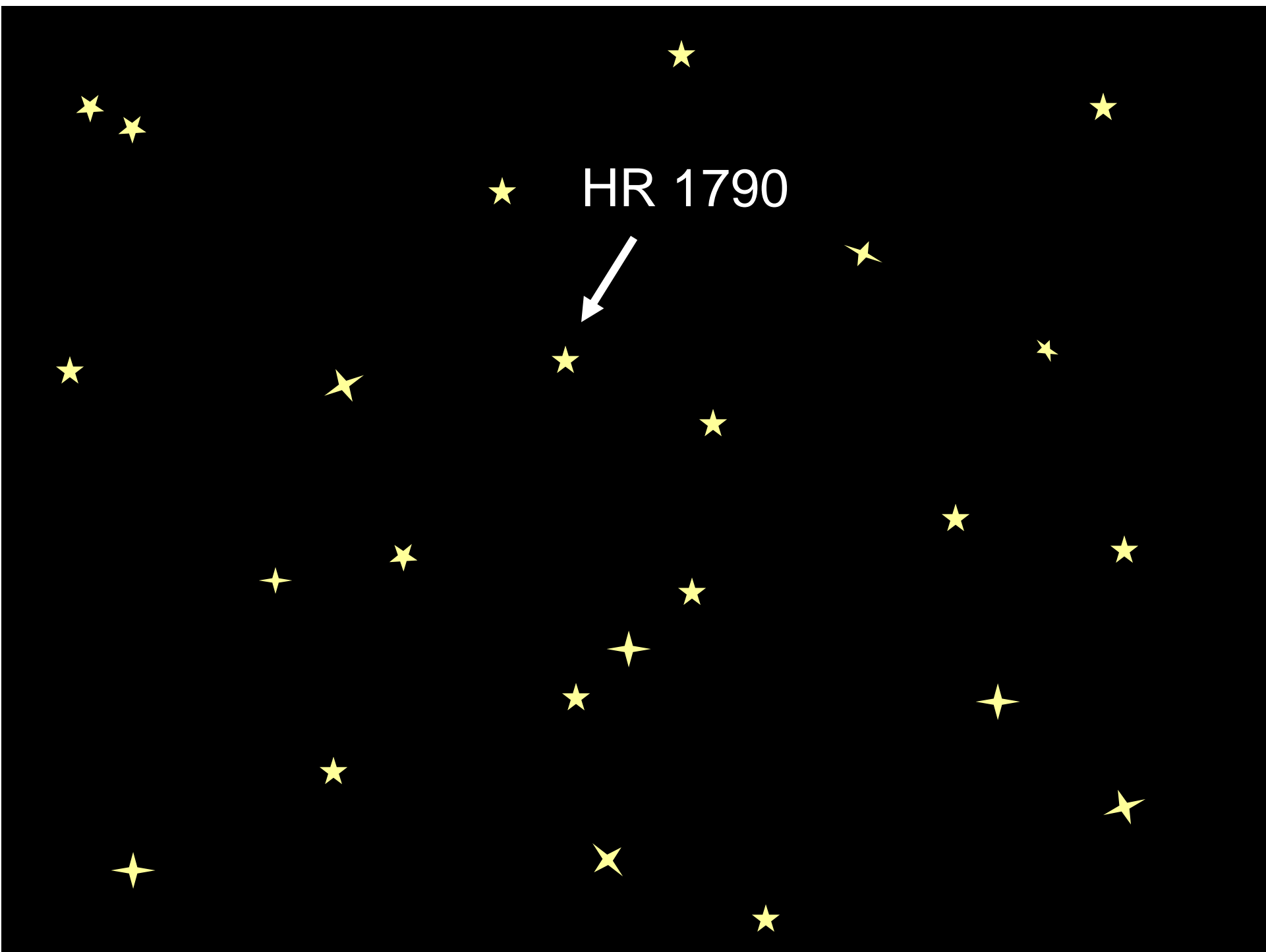
```
392     * Get a 256 color palette, to be used as samples
393     * Incidentally, convert src 32 bit image to dst 8 bit indexed image,
394     * with indices referring to the 256 color palette.
395     * Also incidentally, pack the alpha channel if necessary.
396     */
397     if( txVerbose )
398     {
399         printf("NCC Neural nets..."); fflush(stdout);
400     }
401     pxMip->format = (format == GR_TEXFMT_YIQ_422) ? GR_TEXFMT_P_8 :
402         GR_TEXFMT_AP_88;
403     ncolors = txMipPal256(pxMip, txMip, pxMip->format, 0, 0);
404     if( txVerbose )
405     {
406         printf("%d samples...", ncolors); fflush(stdout);
407     }
408     txMapPal256toYAB((FxU32 *)yabTable, (FxU8 *)map, ncolors, (FxU32 *)pxMip->pal);
409     if( txVerbose )
410     {
411         printf("eMax=(%3ld%3ld%3ld)...eAvg=(%3ld%3ld%3ld)\n",
412             errG, errR, errB,
413             totG/ncolors, totR/ncolors, totB/ncolors
```

**Function Called Here**

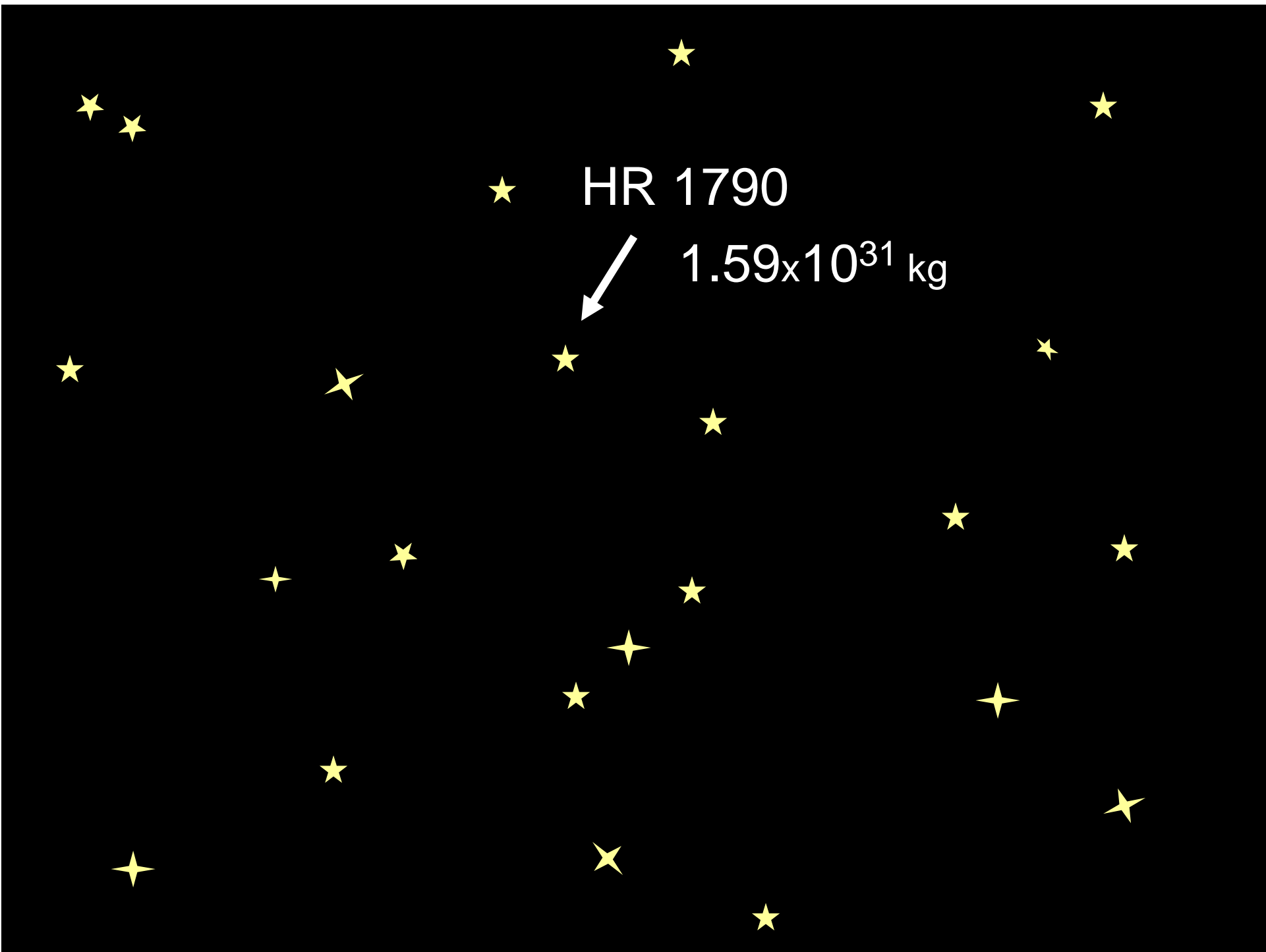






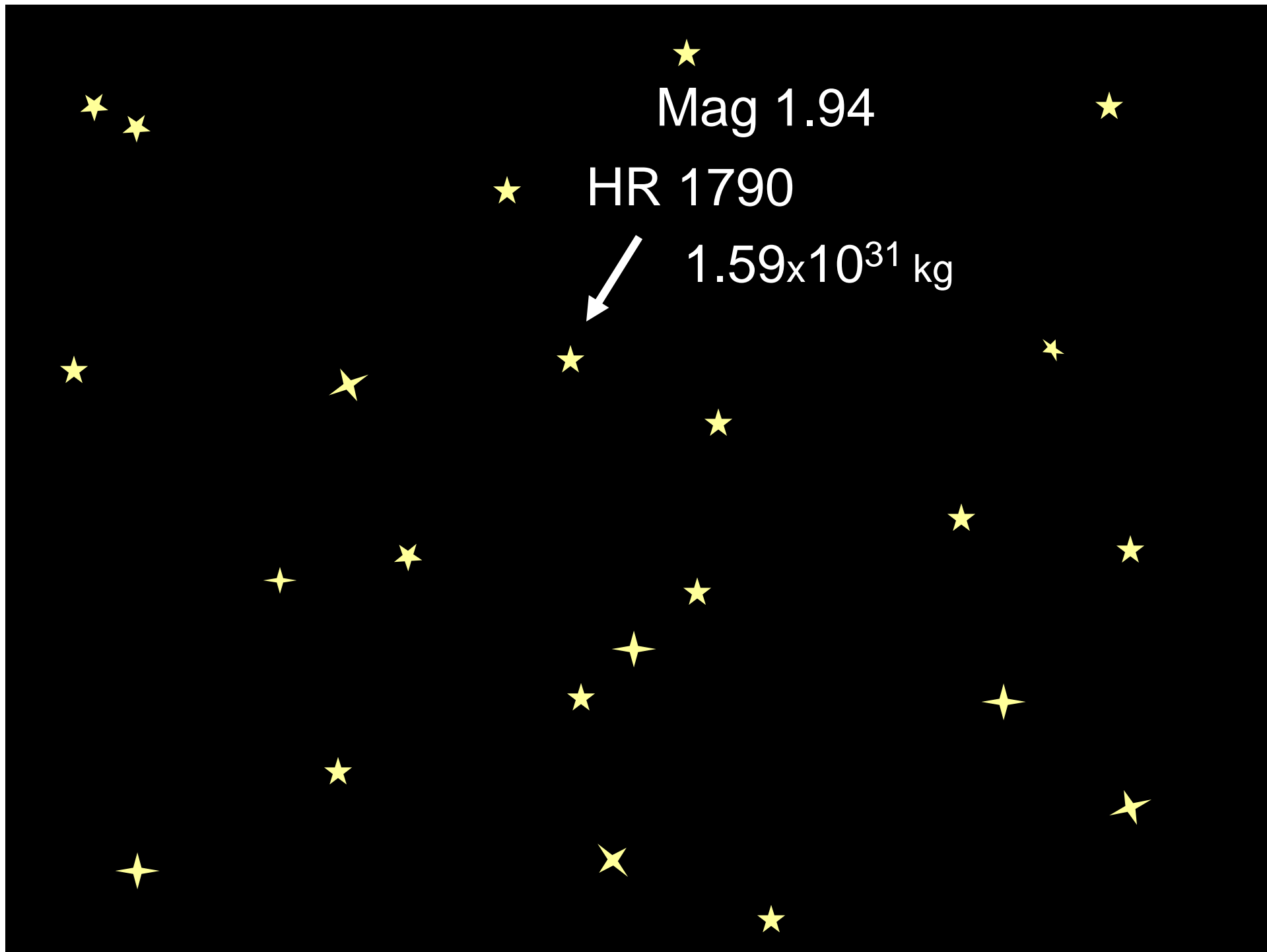


HR 1790



HR 1790

$1.59 \times 10^{31} \text{ kg}$

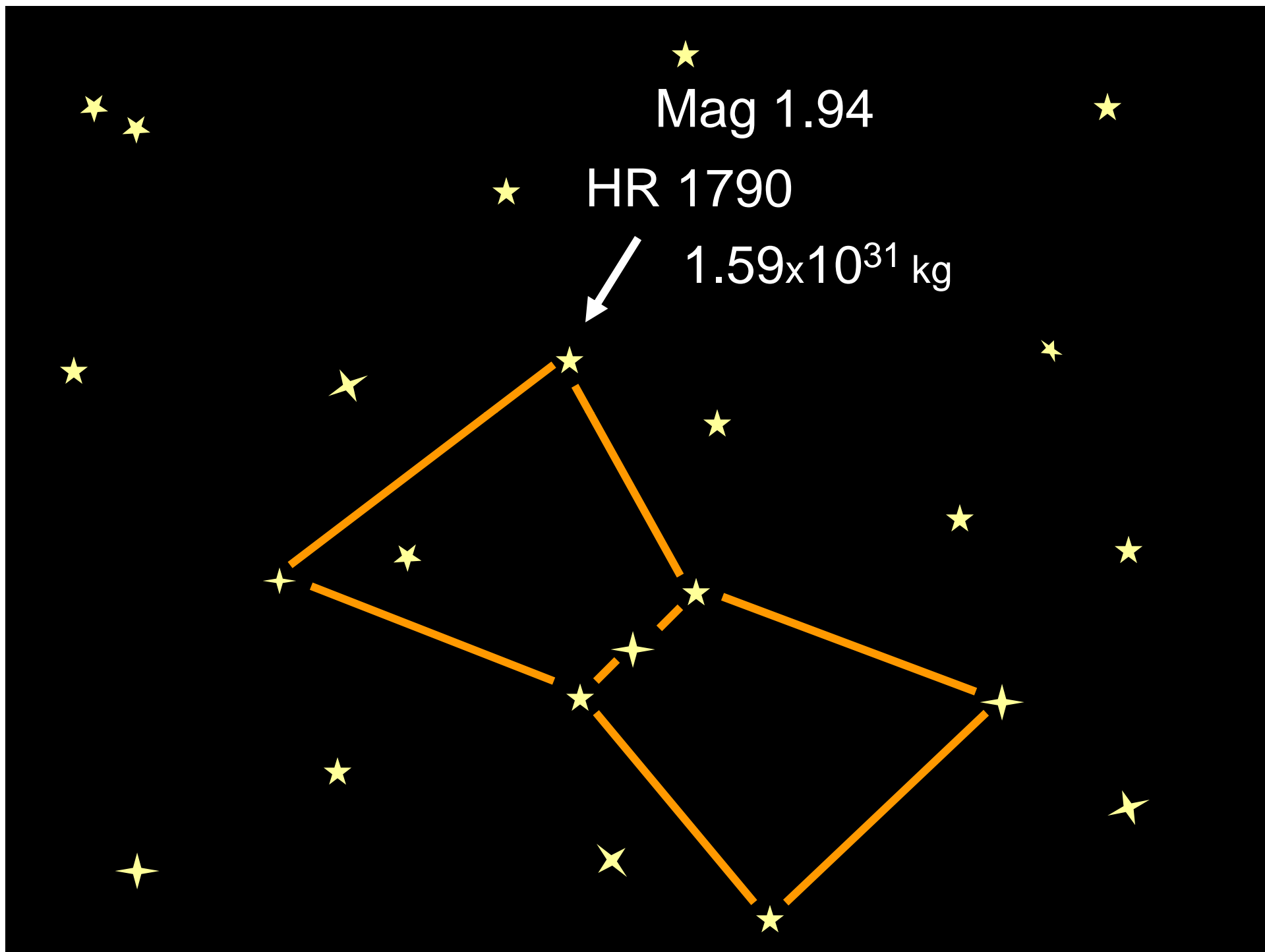


★  
Mag 1.94

★ HR 1790

1.59x10<sup>31</sup> kg

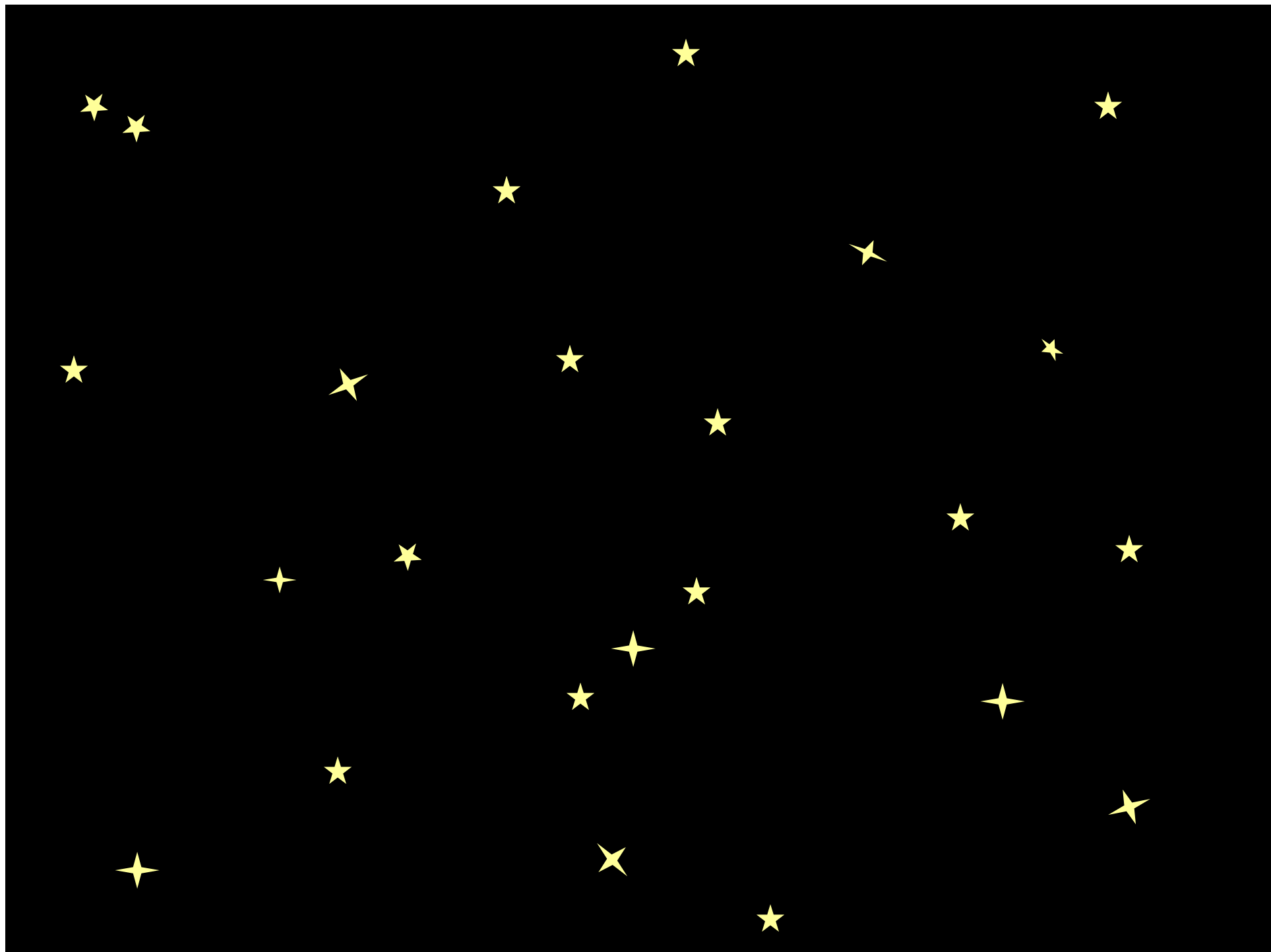


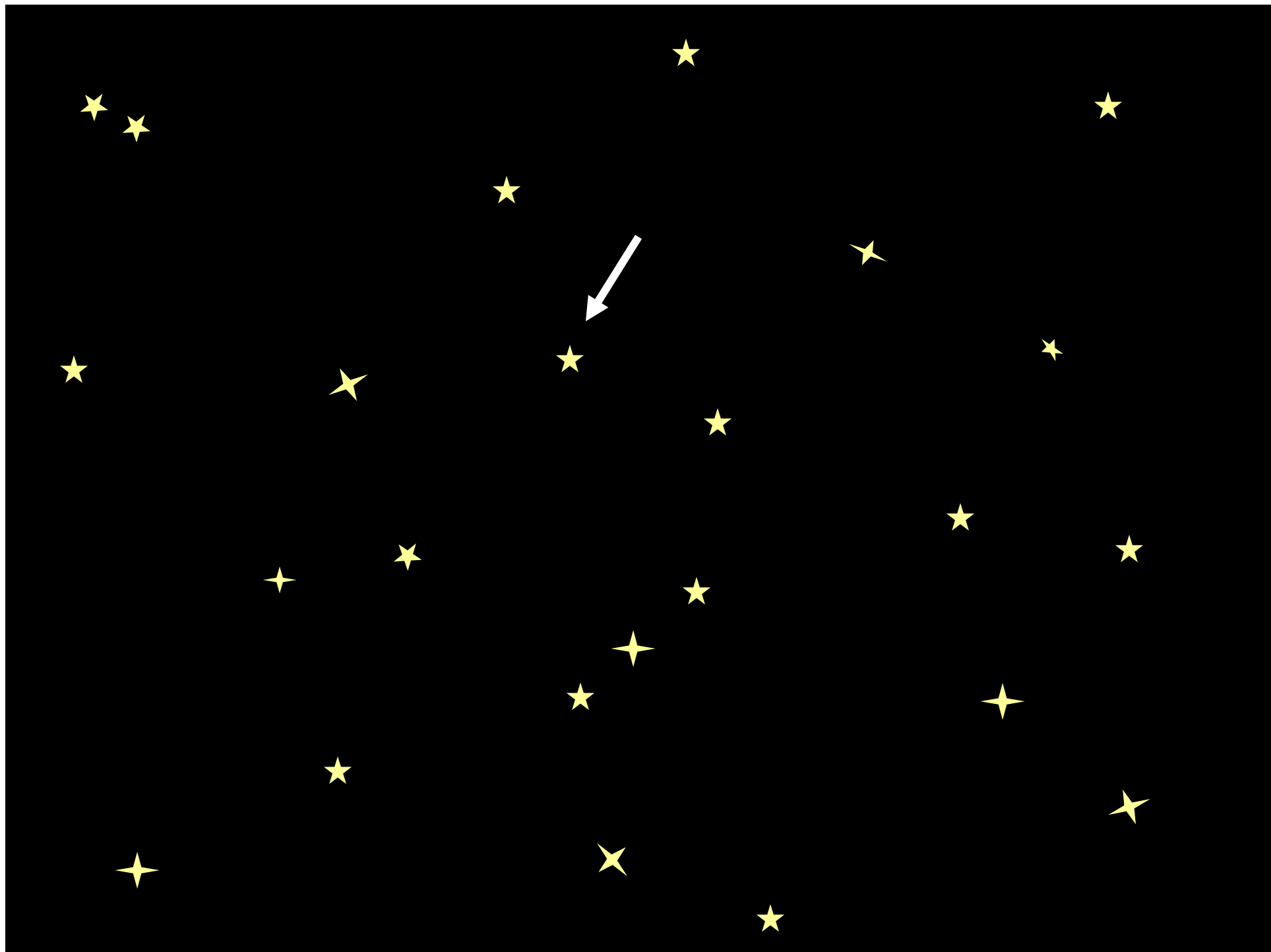


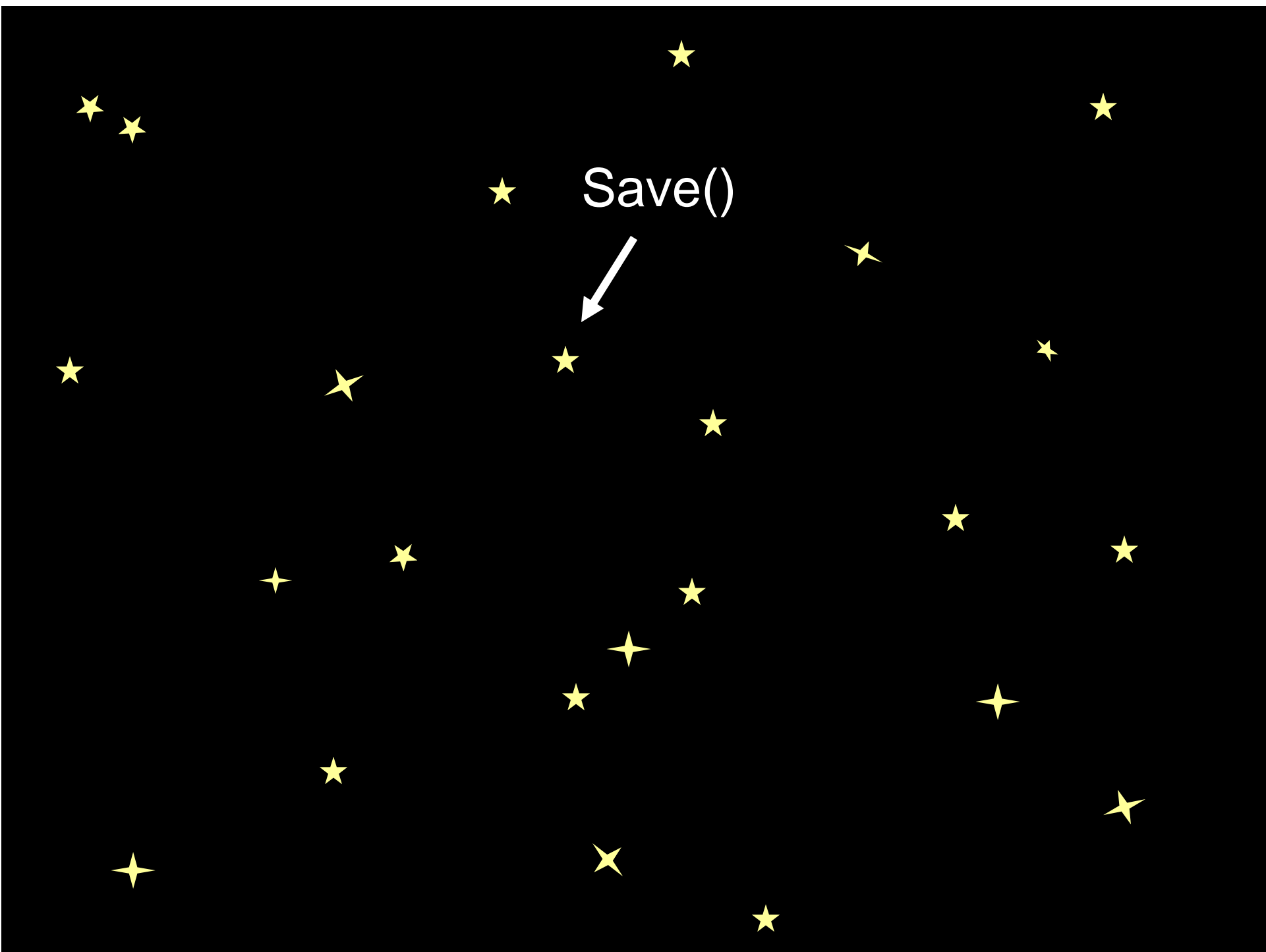
Mag 1.94

HR 1790

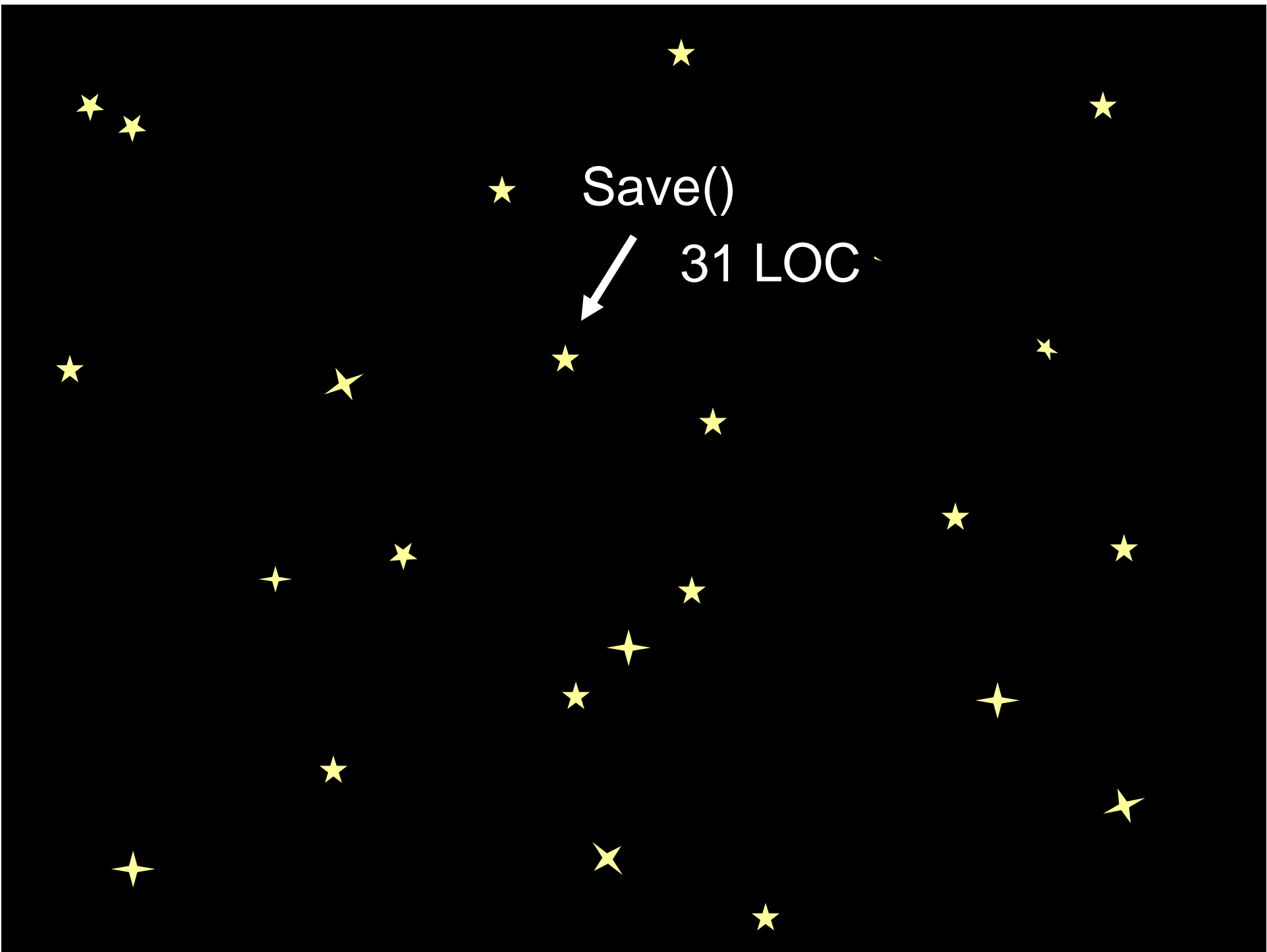
$1.59 \times 10^{31}$  kg











Save()

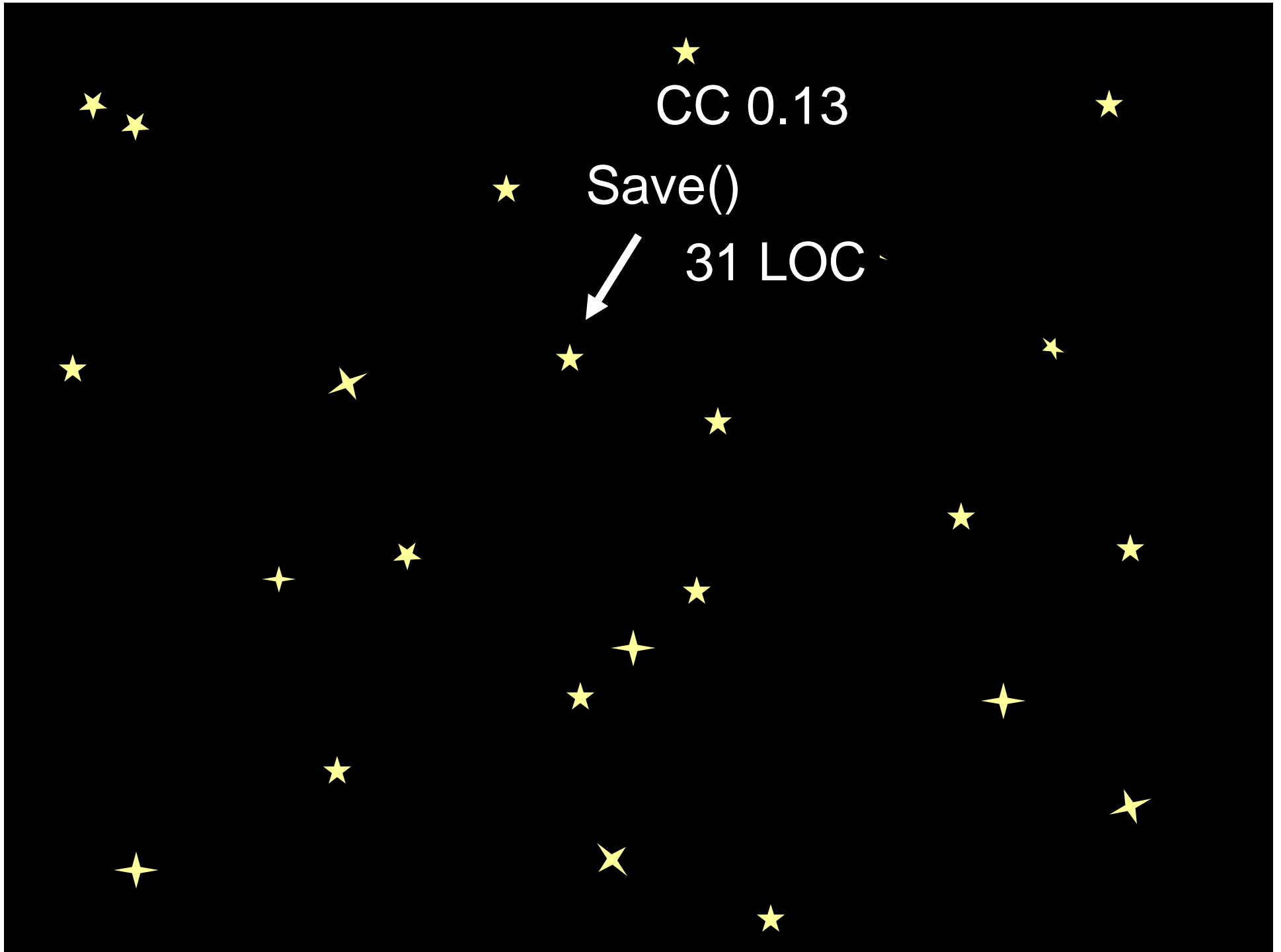
31 LOC

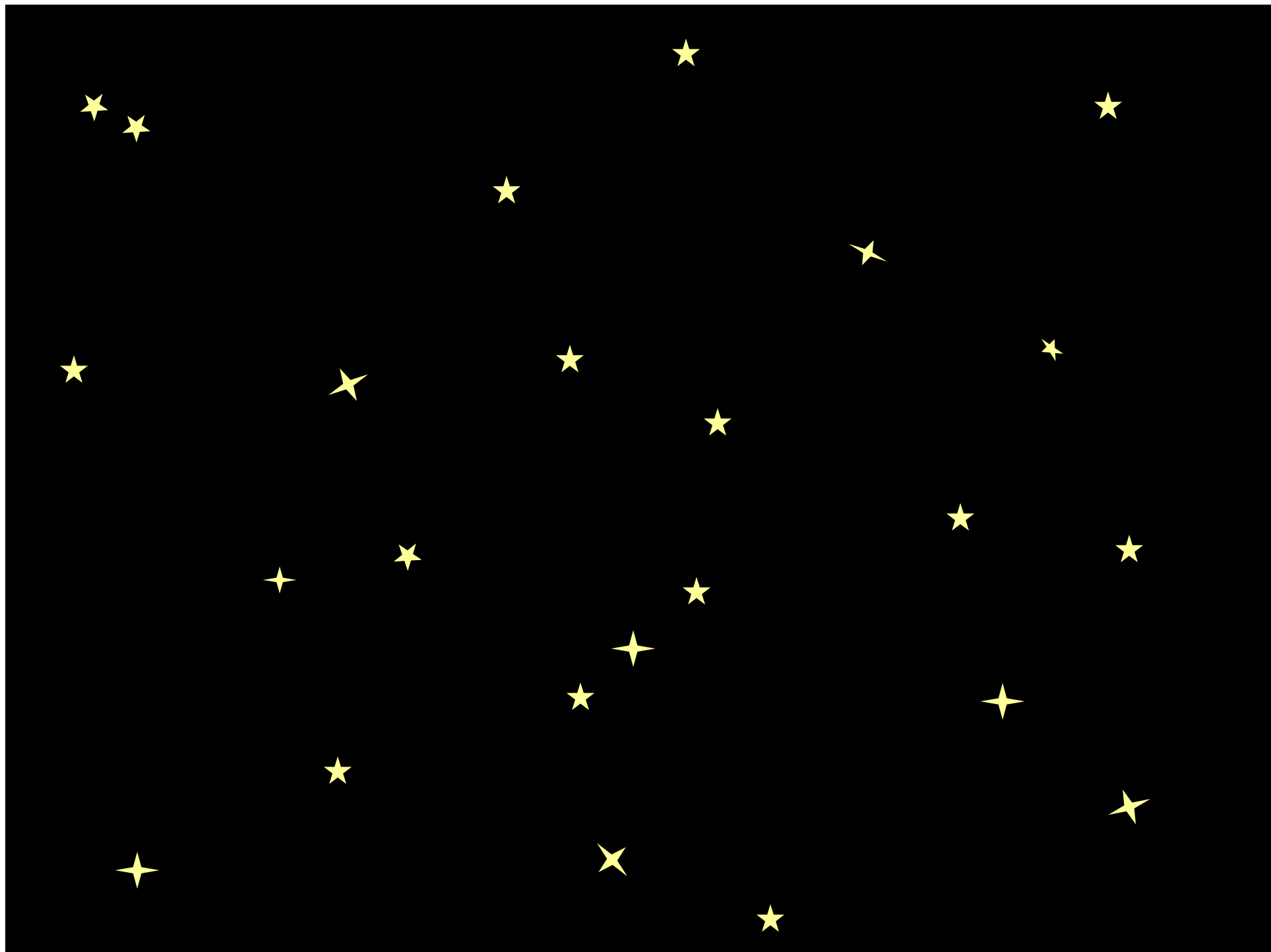


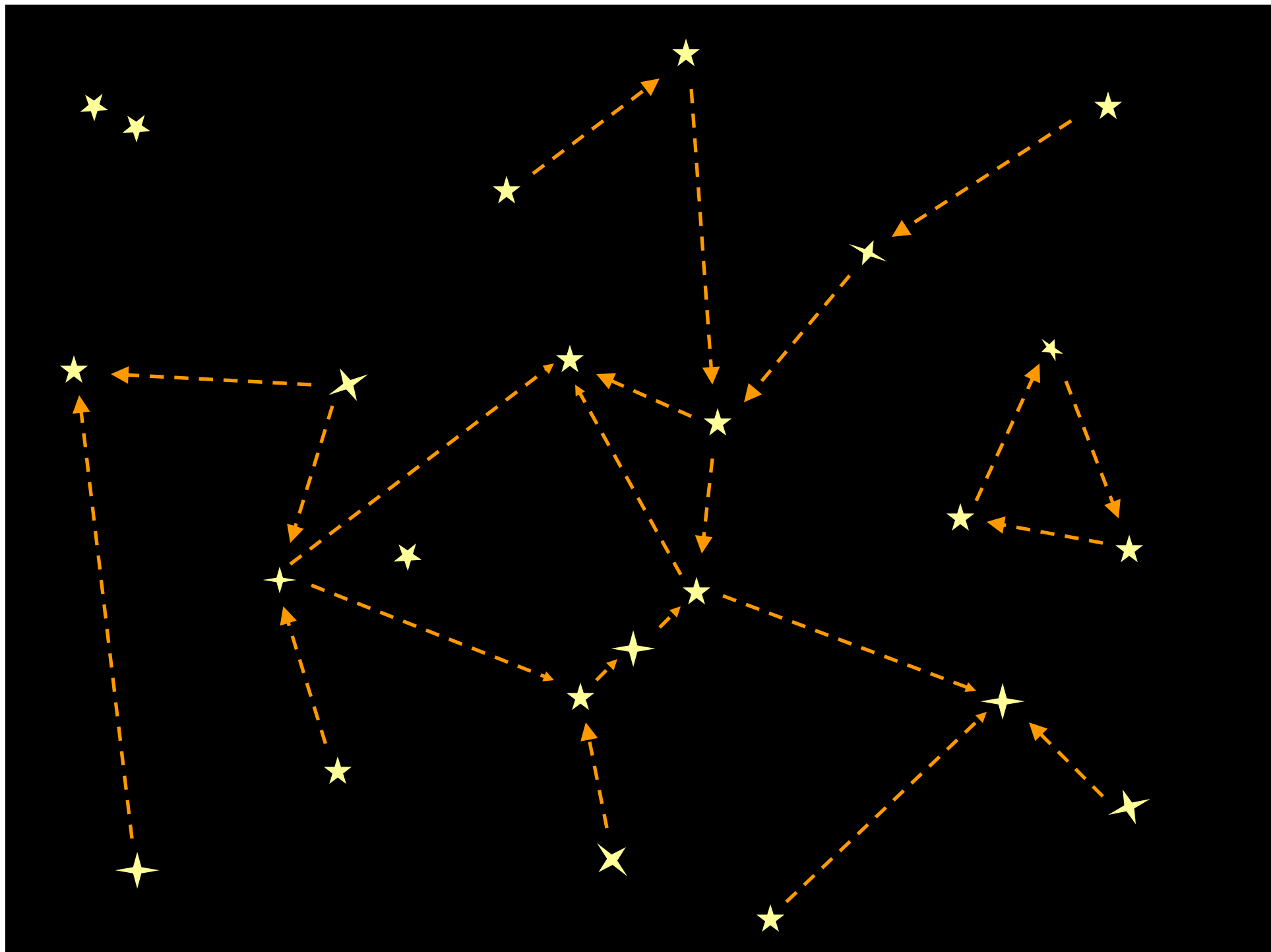
★  
CC 0.13

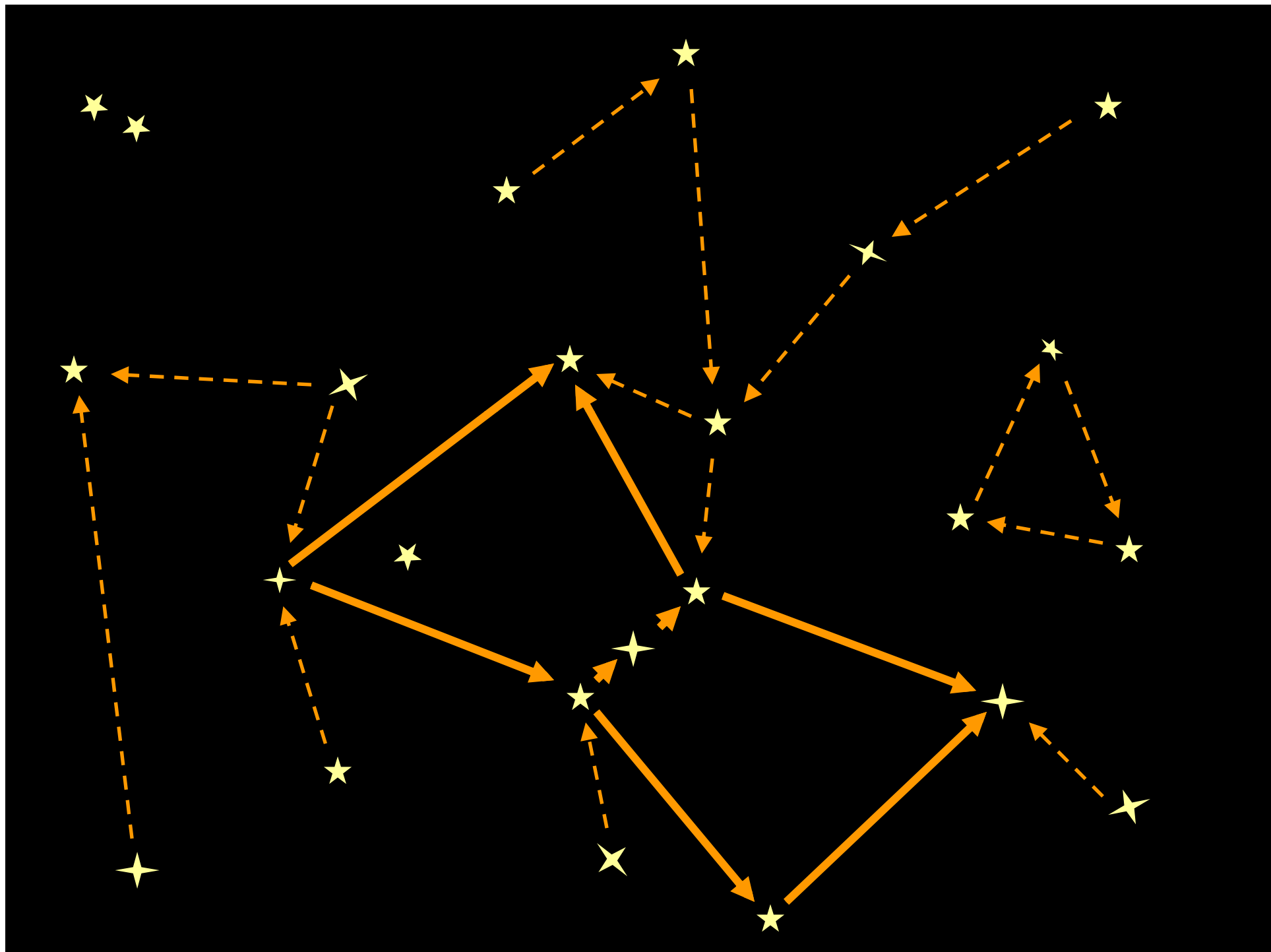
★ Save()

31 LOC









# Our Search Engine: Portfolio

Our Contribution, Two Search Models:

- 1) Navigation Model for Software
- 2) Association Model for Software

# Navigation Model

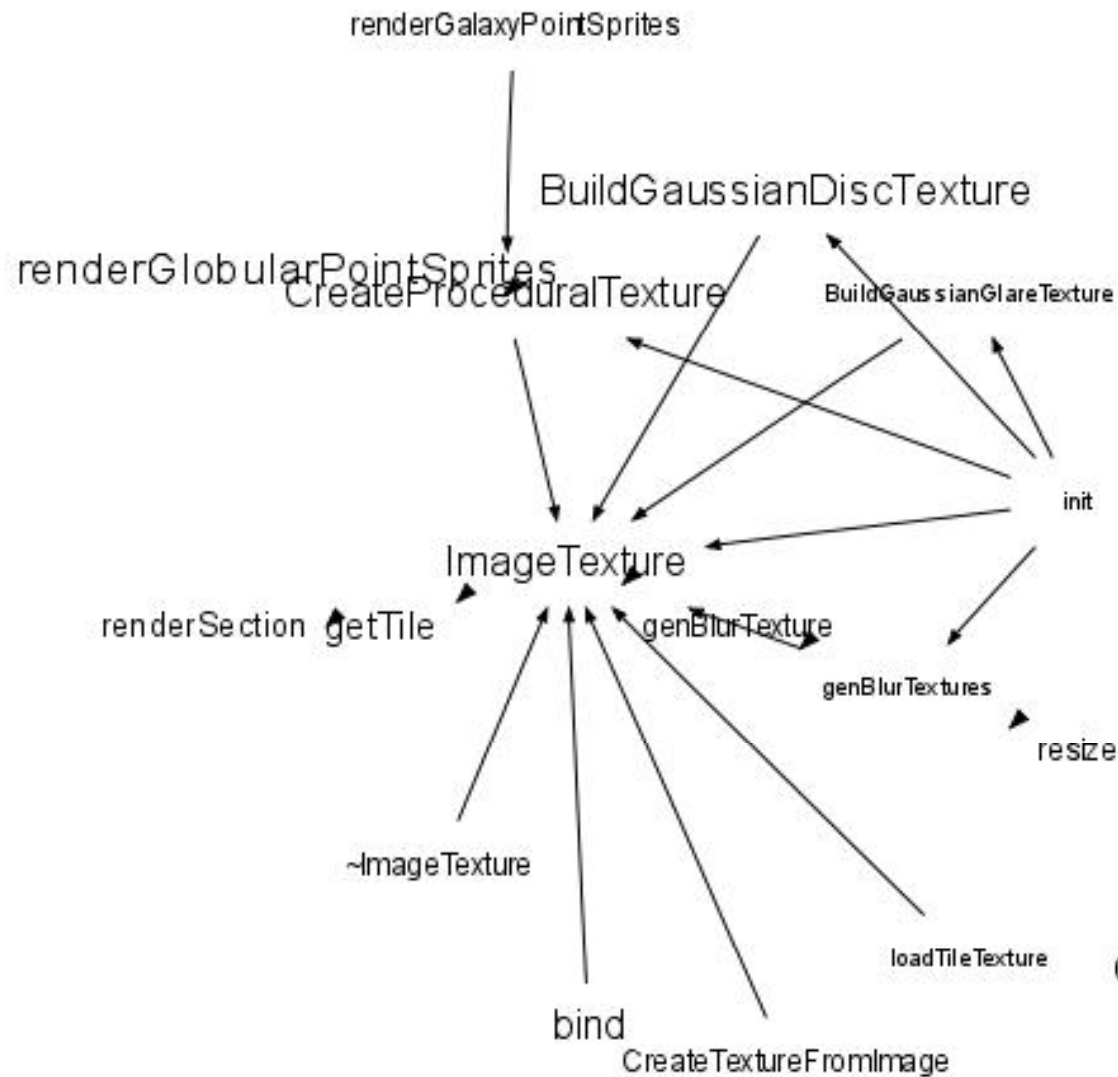


# Navigation Model - PageRank

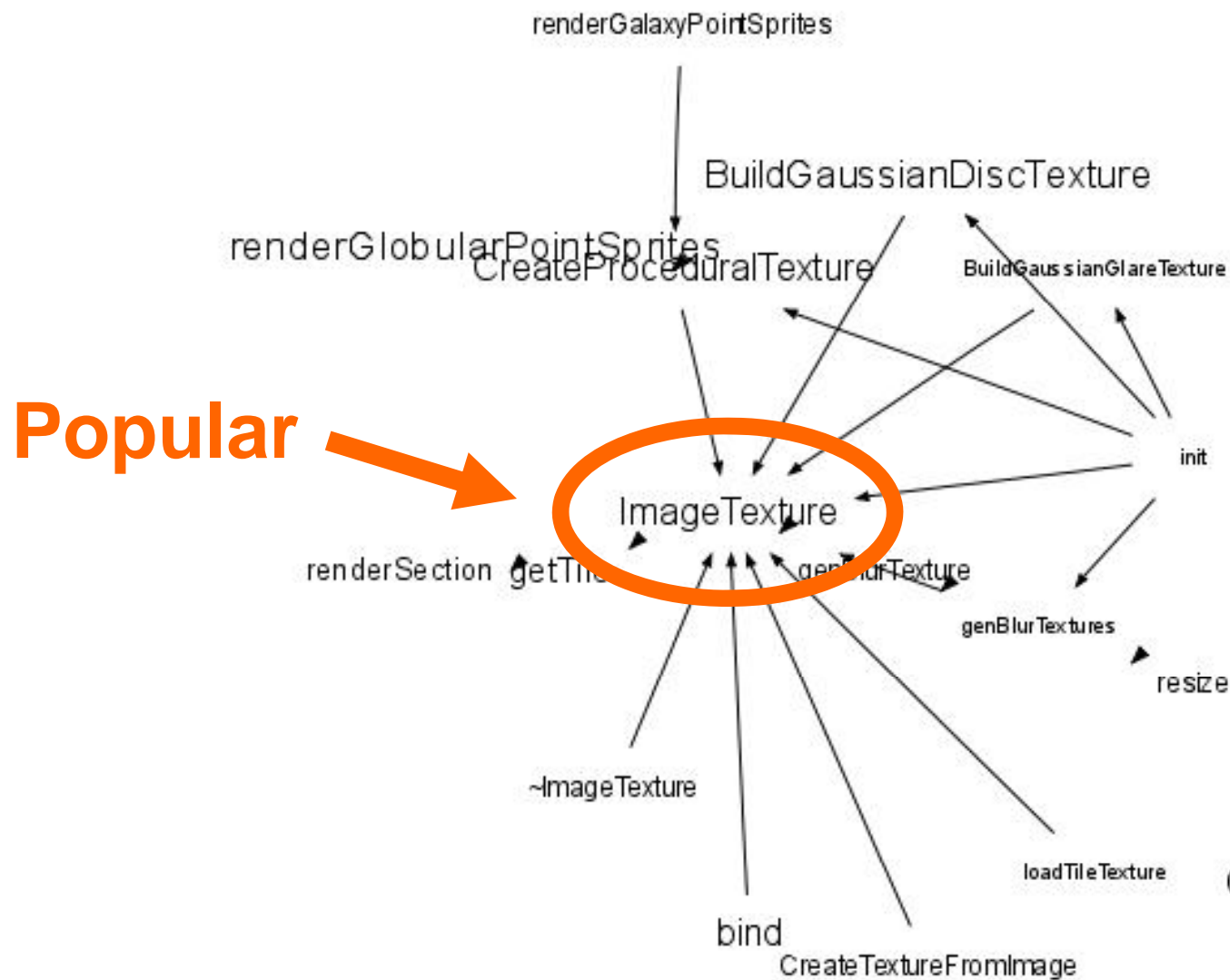




# Navigation Model - PageRank



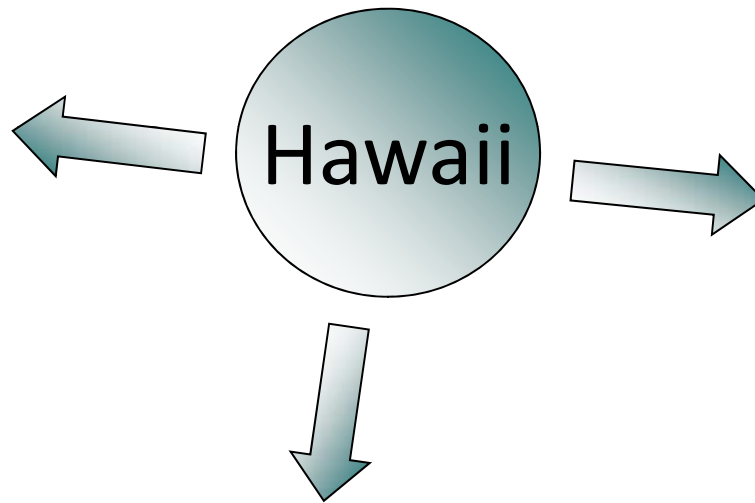
# Navigation Model - PageRank



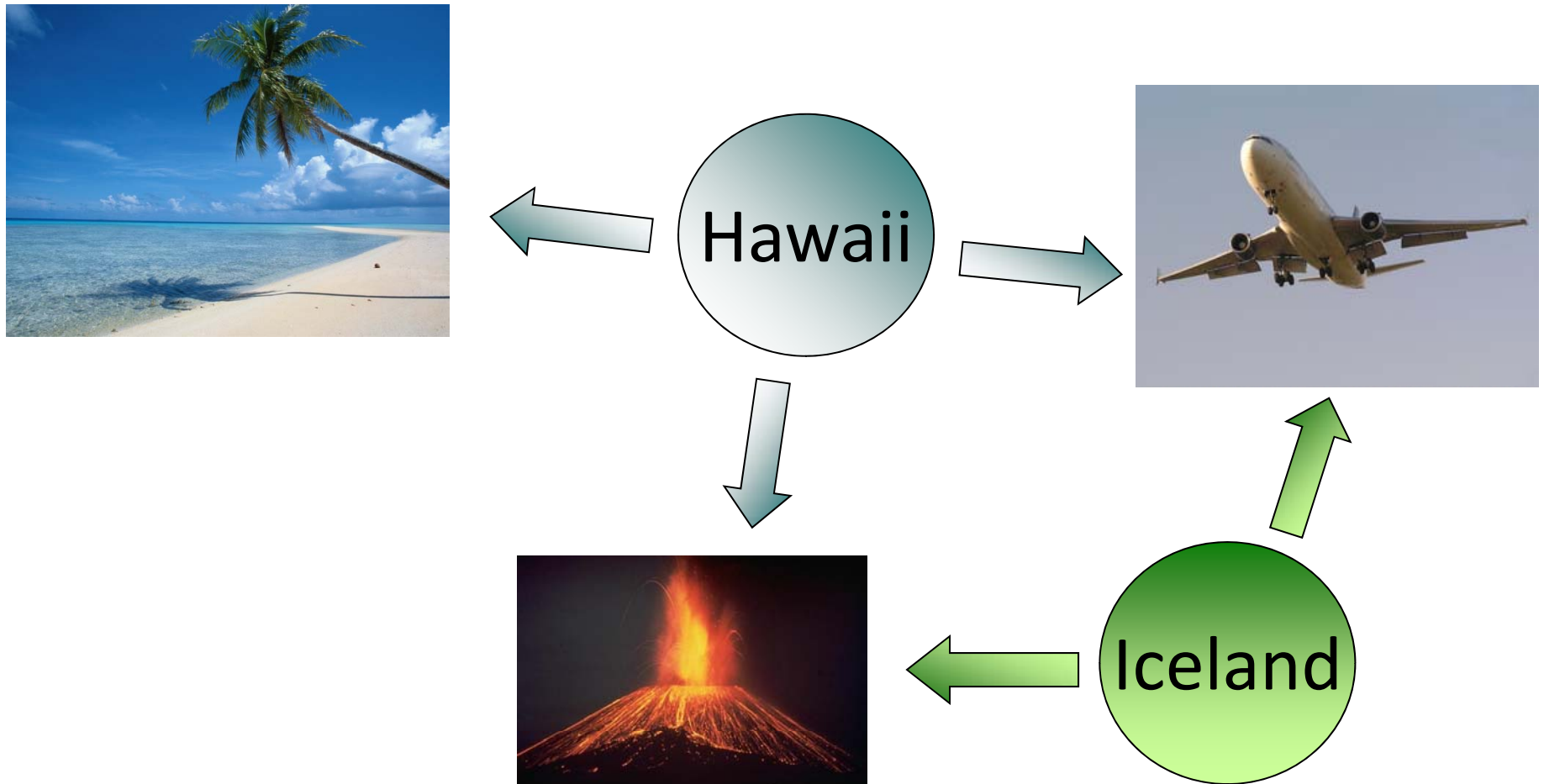
# Spreading Activation in Associative Networks



# Spreading Activation in Associative Networks



# Spreading Activation in Associative Networks



# In Portfolio: Function calls are associations

`initRenderer`

`LoadTextureFromFile`

`GlareTexture`

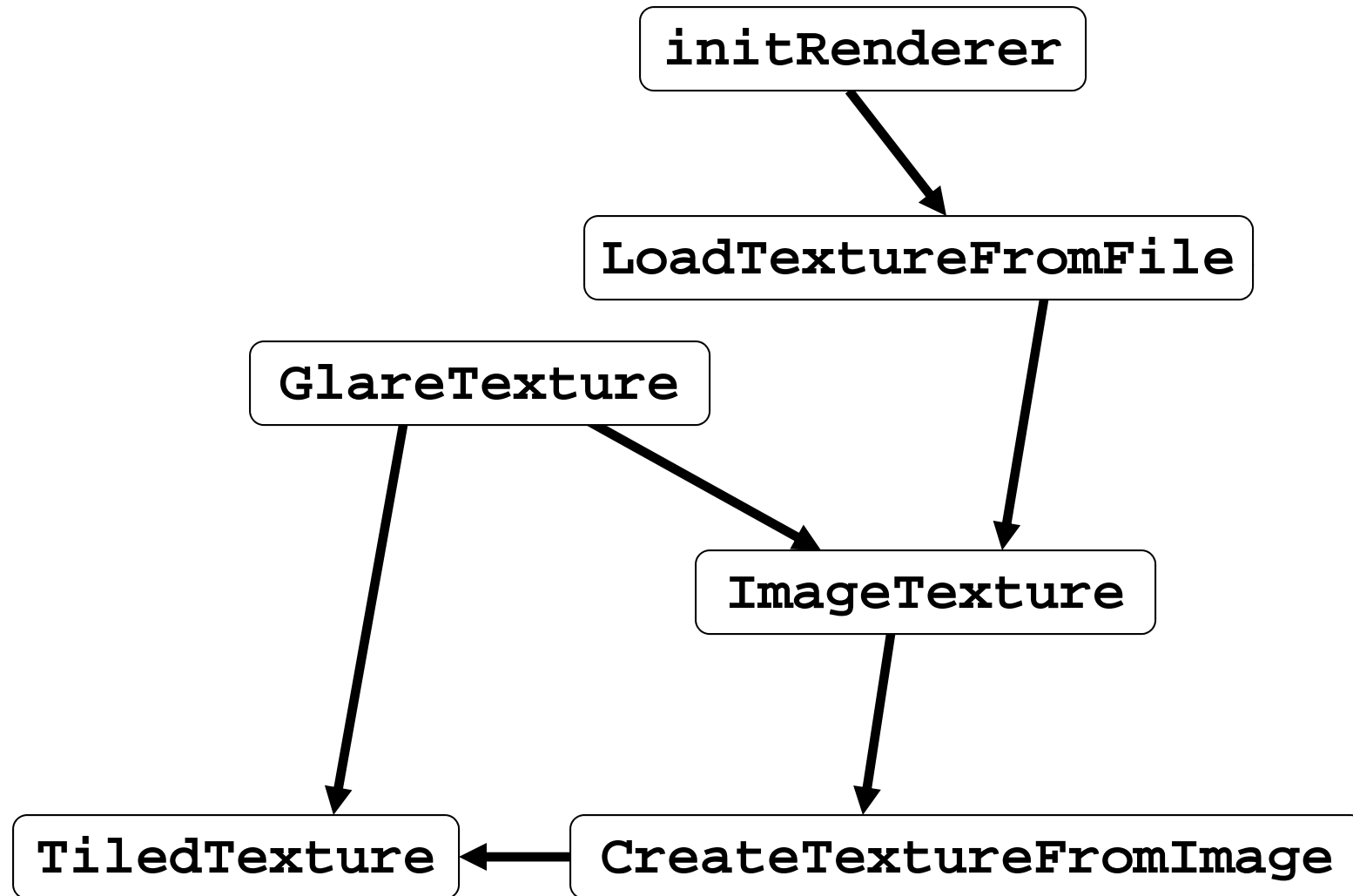
`ImageTexture`

`TiledTexture`

`CreateTextureFromImage`

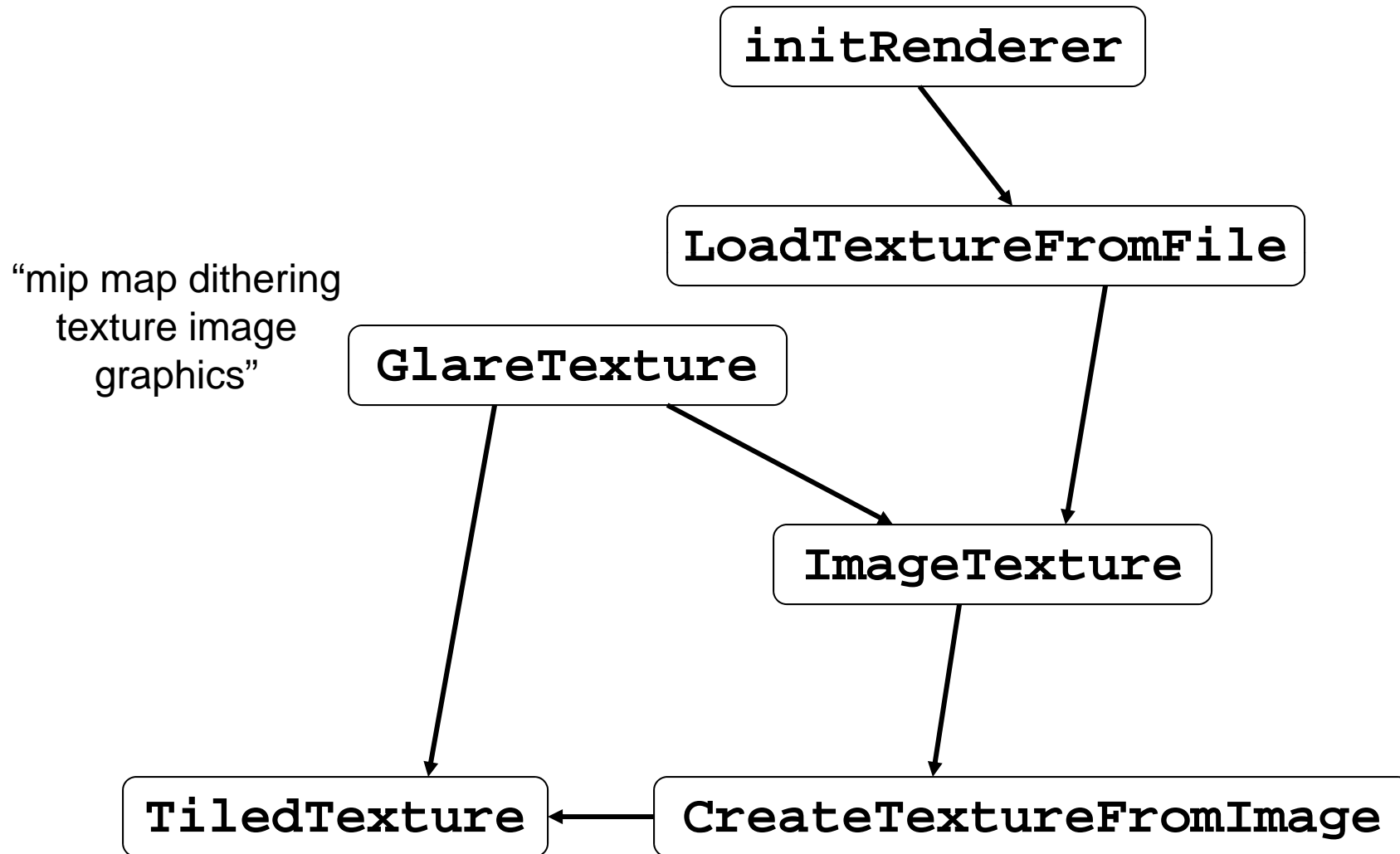
Functions from Celestia: <http://www.shatters.net/celestia/>

# In Portfolio: Function calls are associations



Functions from Celestia: <http://www.shatters.net/celestia/>

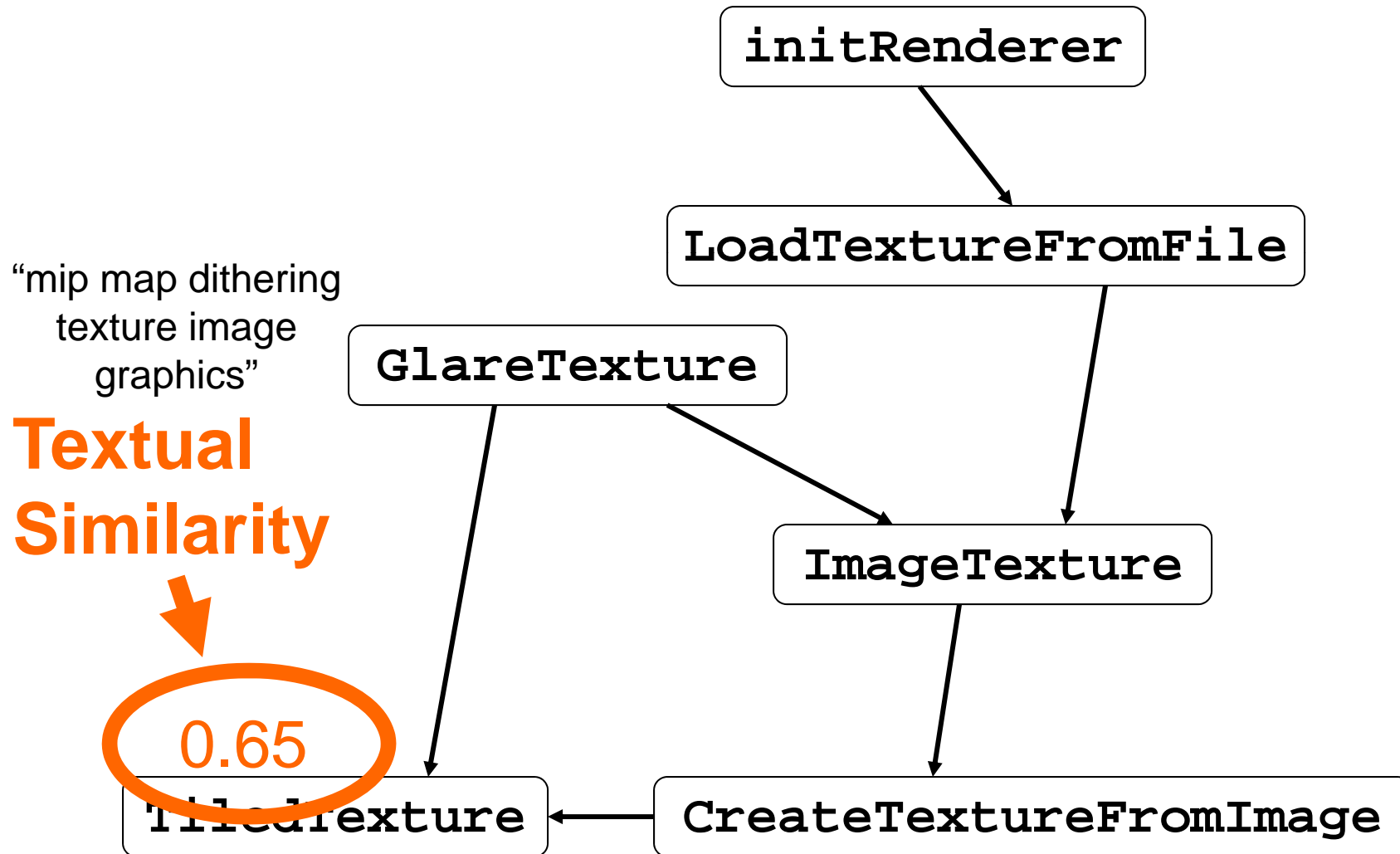
# Association Model – Spreading Activation



Functions from Celestia: <http://www.shatters.net/celestia/>

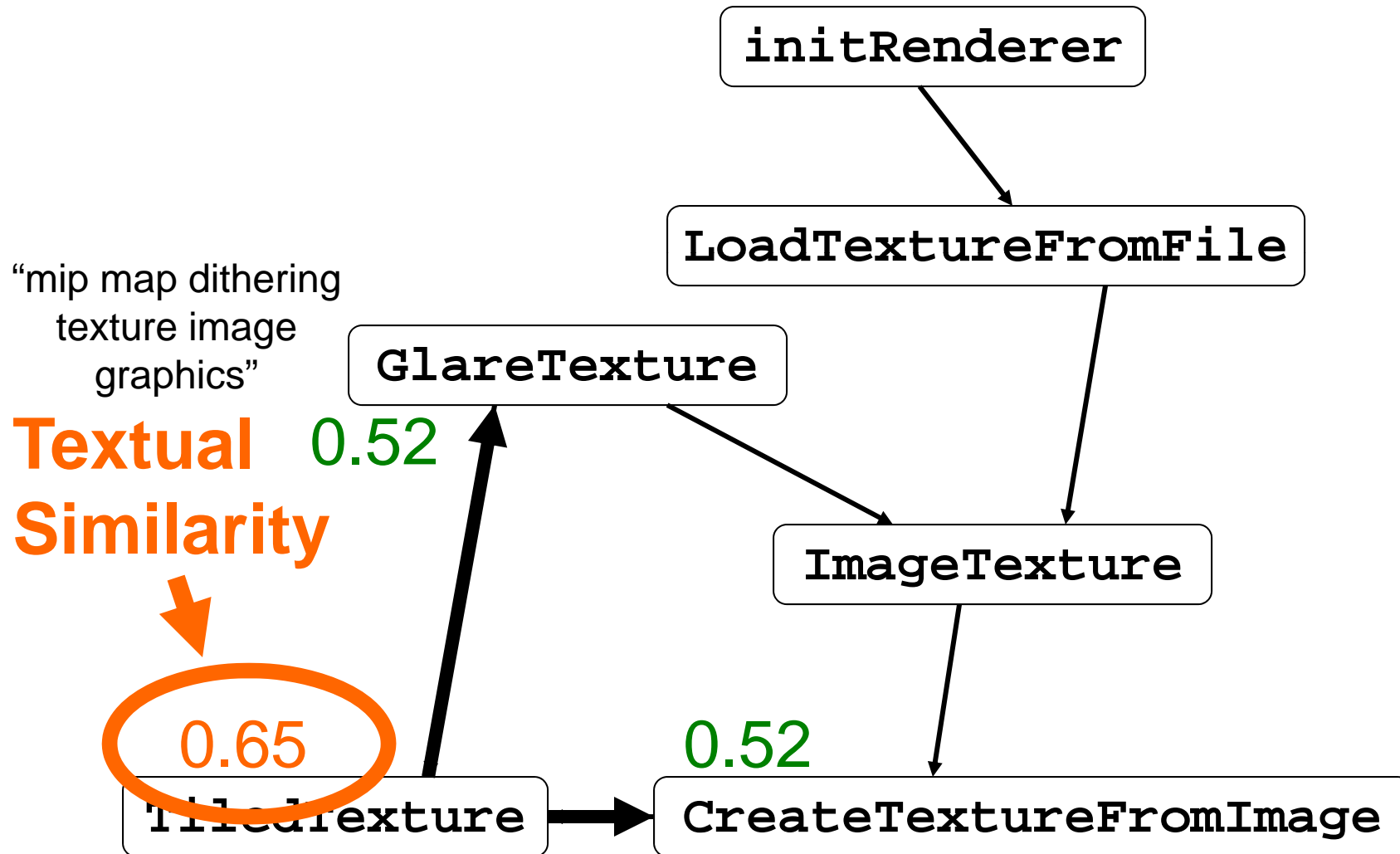


# Association Model – Spreading Activation



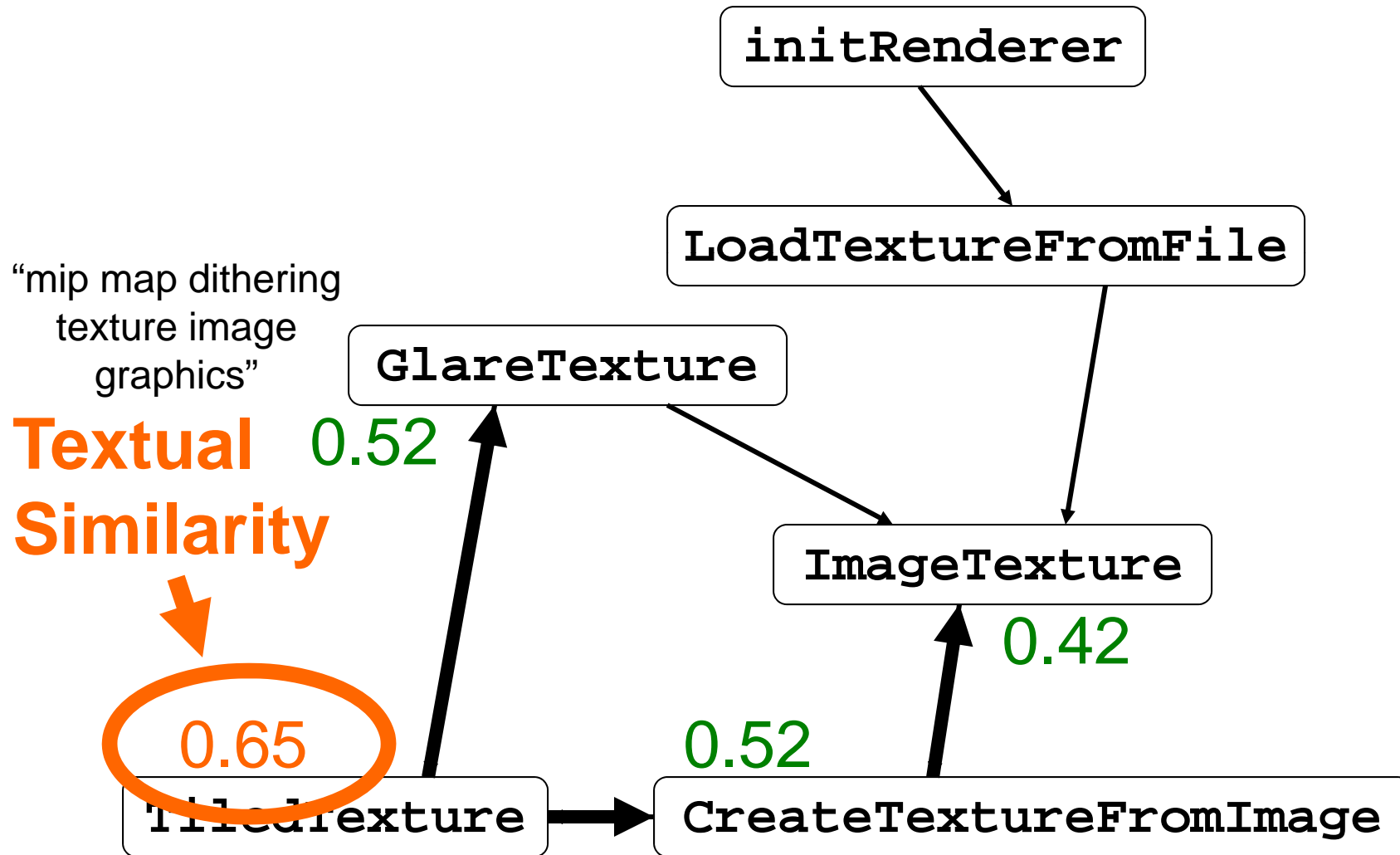
Functions from Celestia: <http://www.shatters.net/celestia/>

# Association Model – Spreading Activation



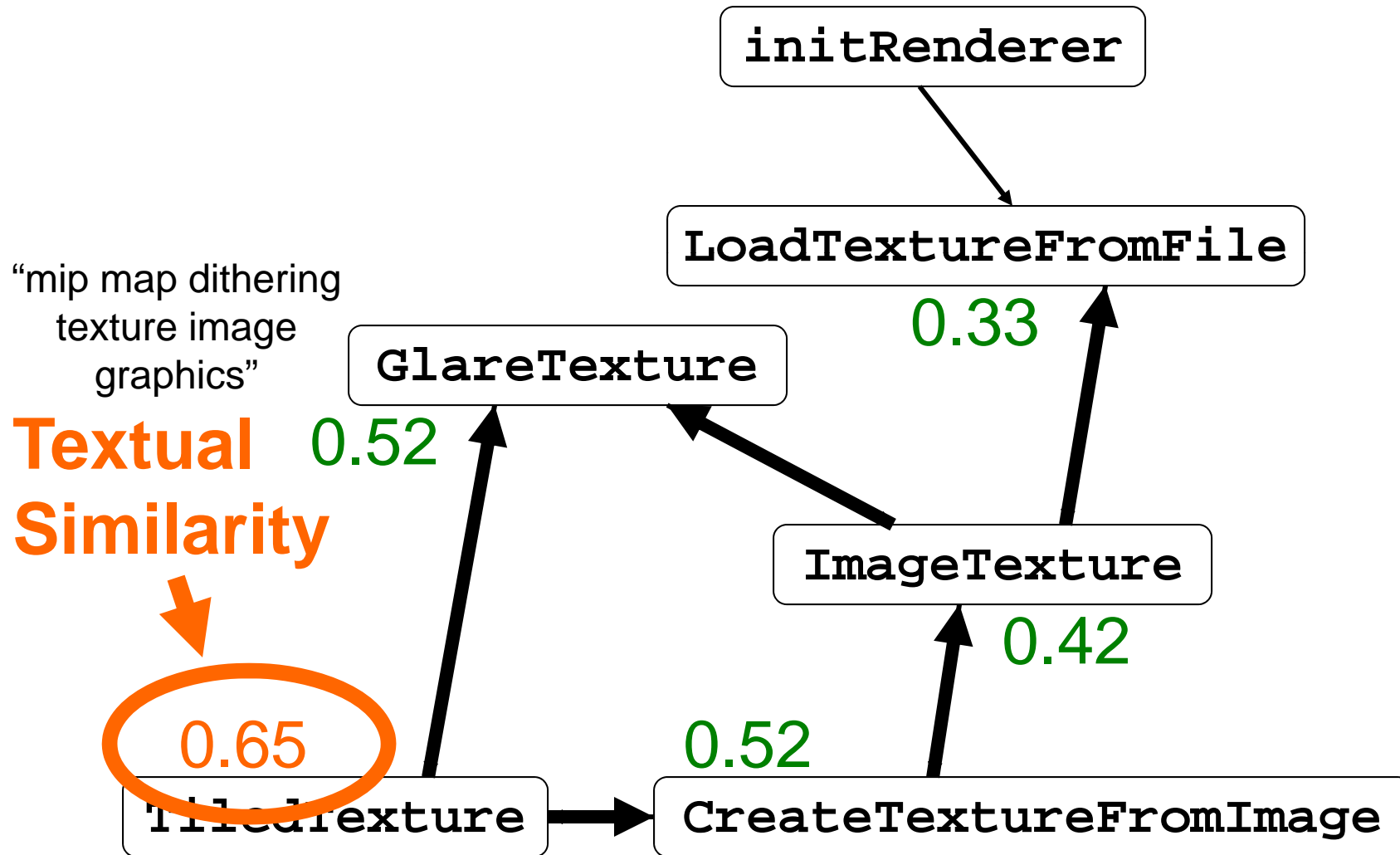
Functions from Celestia: <http://www.shatters.net/celestia/>

# Association Model – Spreading Activation



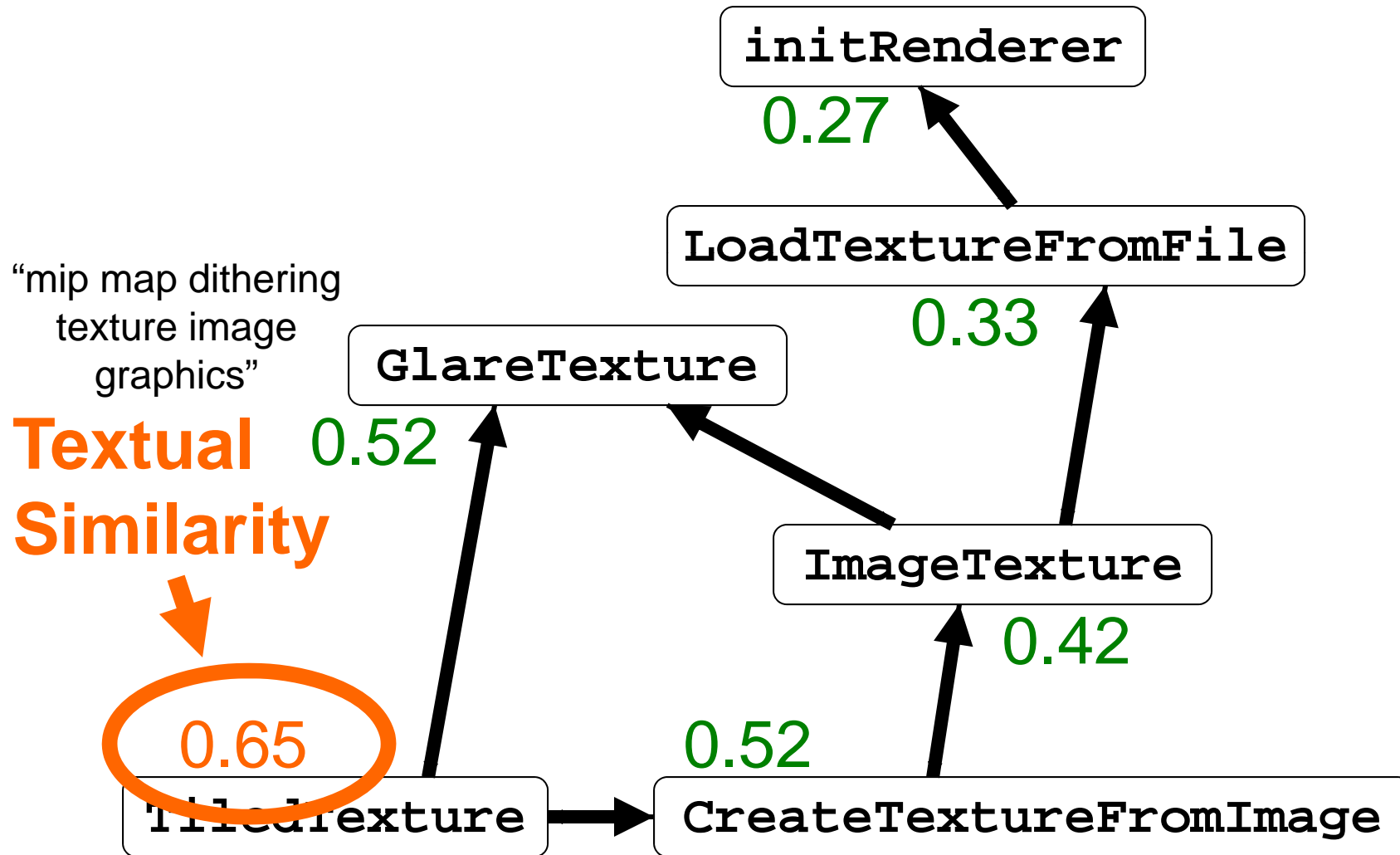
Functions from Celestia: <http://www.shatters.net/celestia/>

# Association Model – Spreading Activation



Functions from Celestia: <http://www.shatters.net/celestia/>

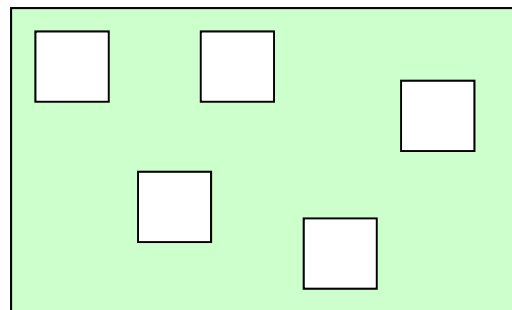
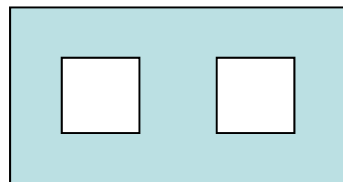
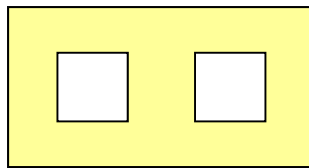
# Association Model – Spreading Activation



Functions from Celestia: <http://www.shatters.net/celestia/>

---

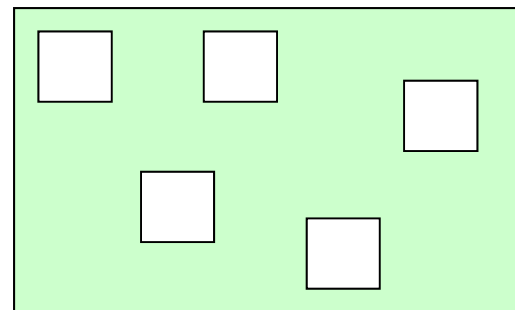
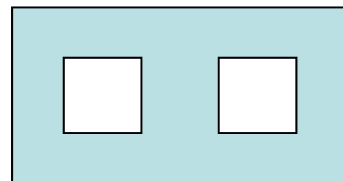
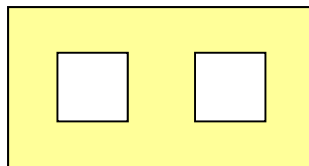


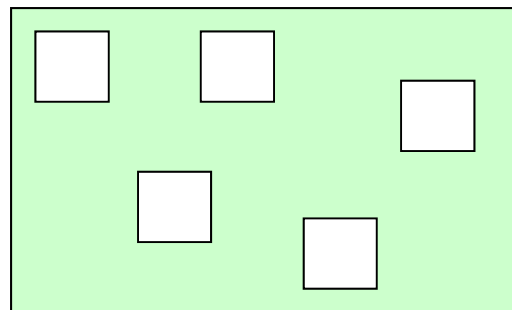
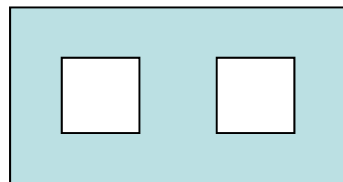
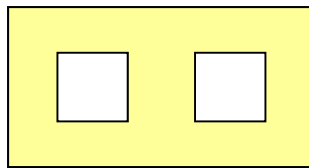
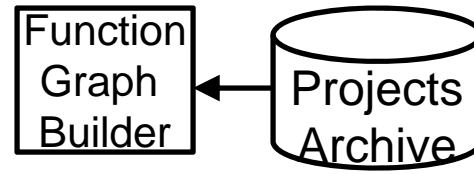


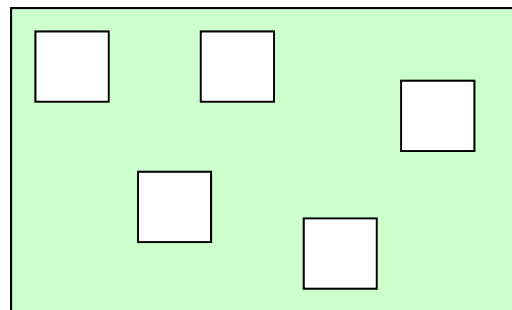
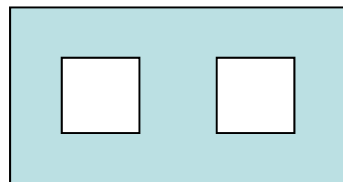
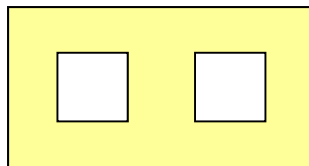
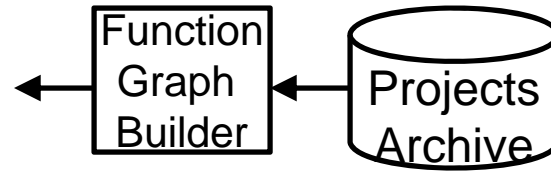


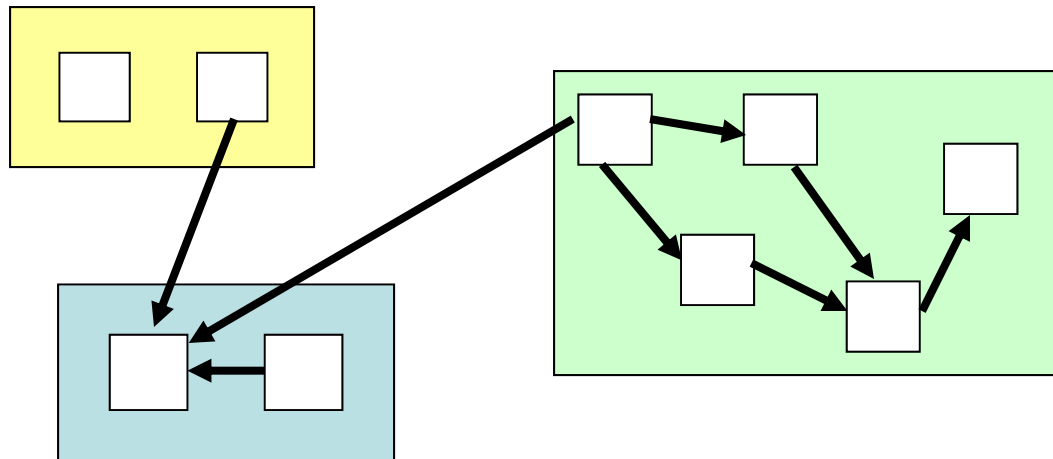
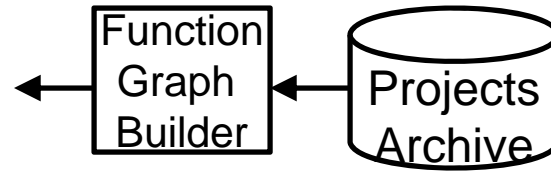
Function  
Graph  
Builder

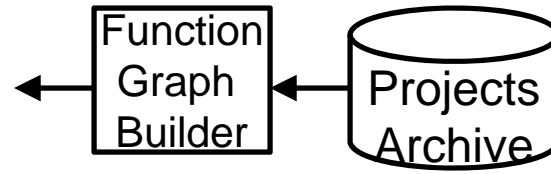
Projects  
Archive

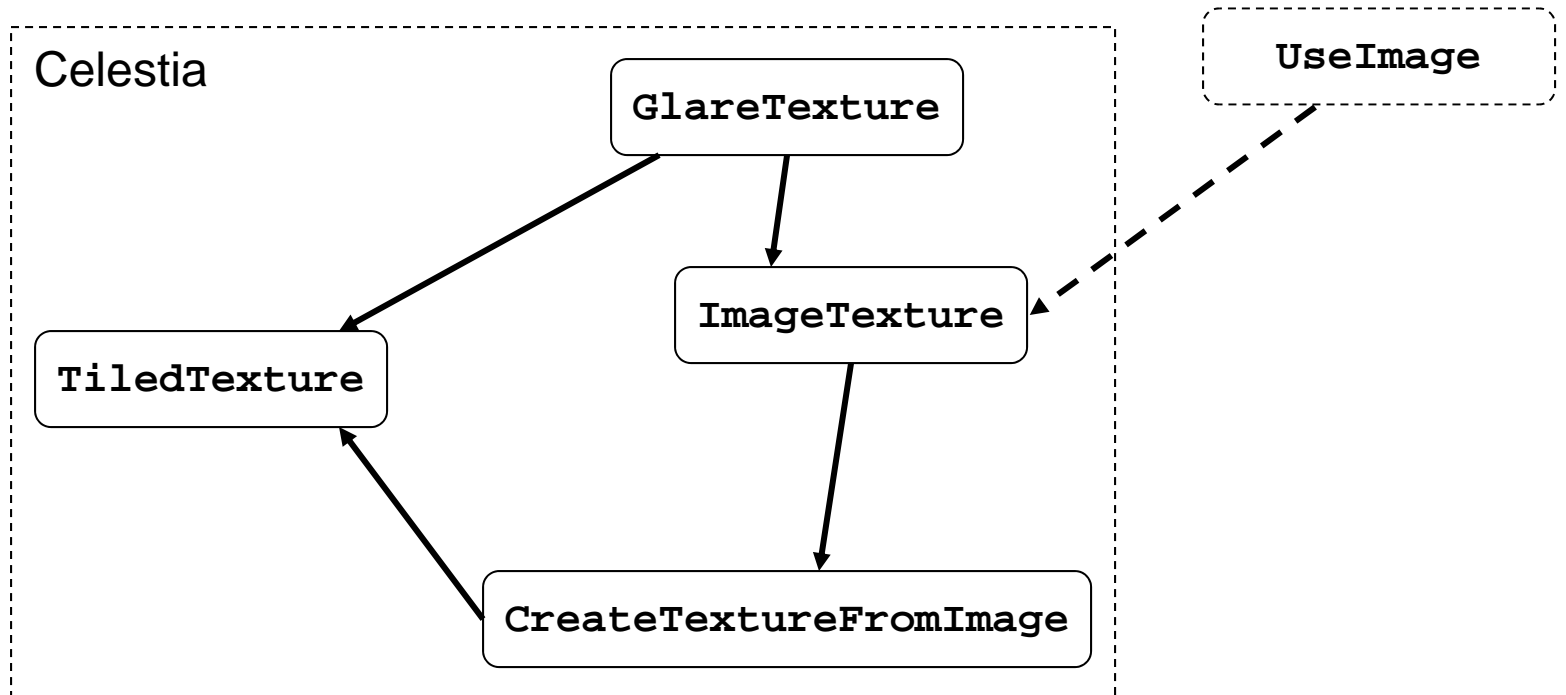
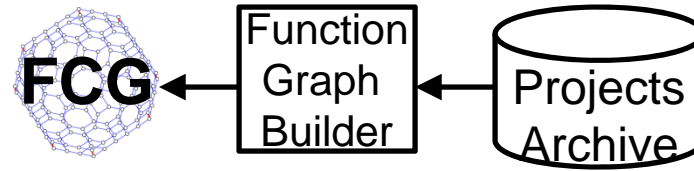


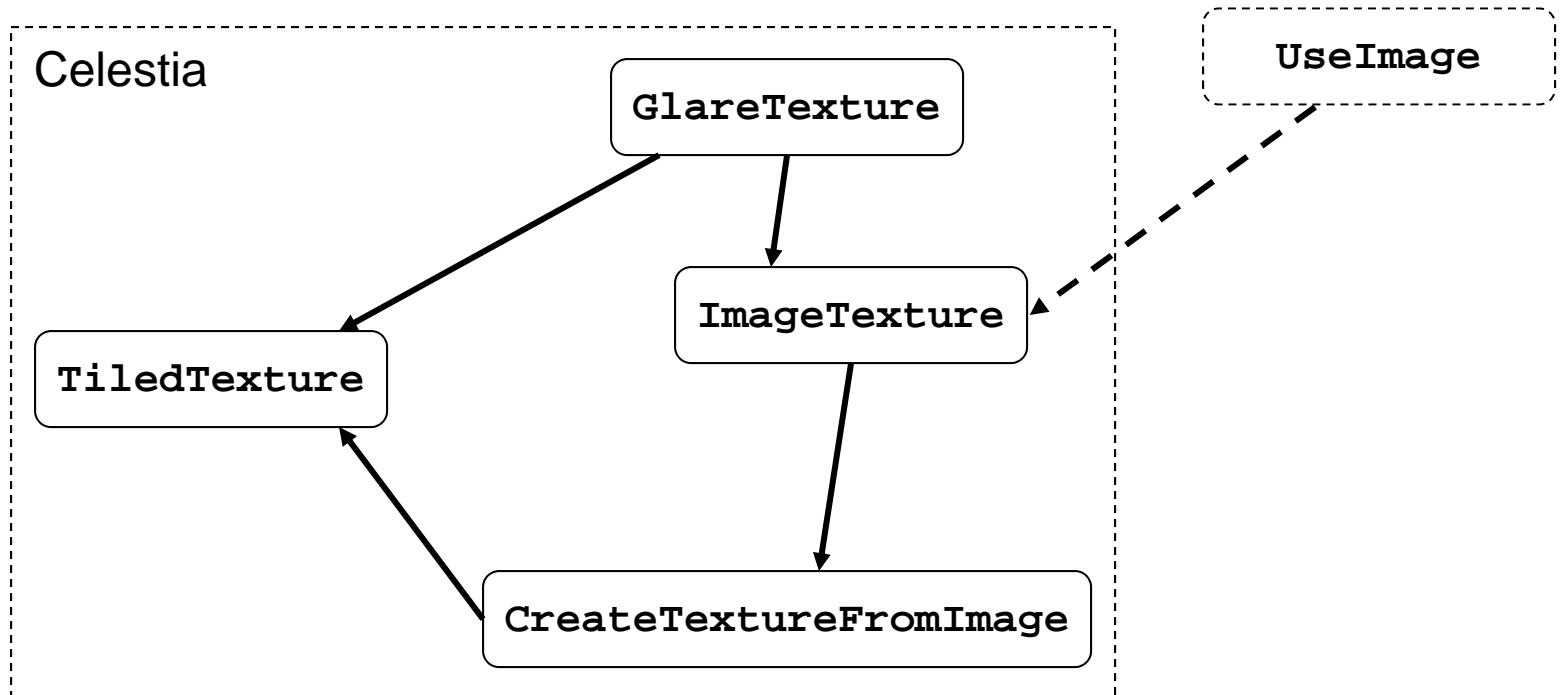
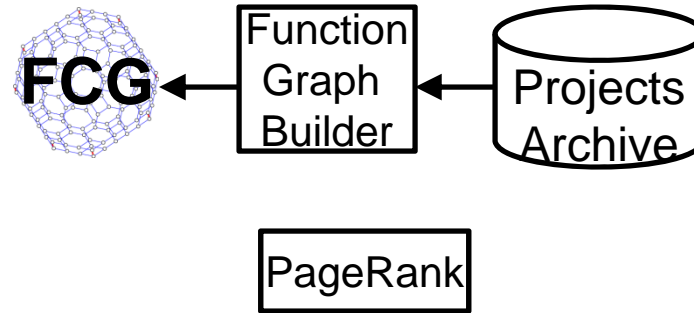


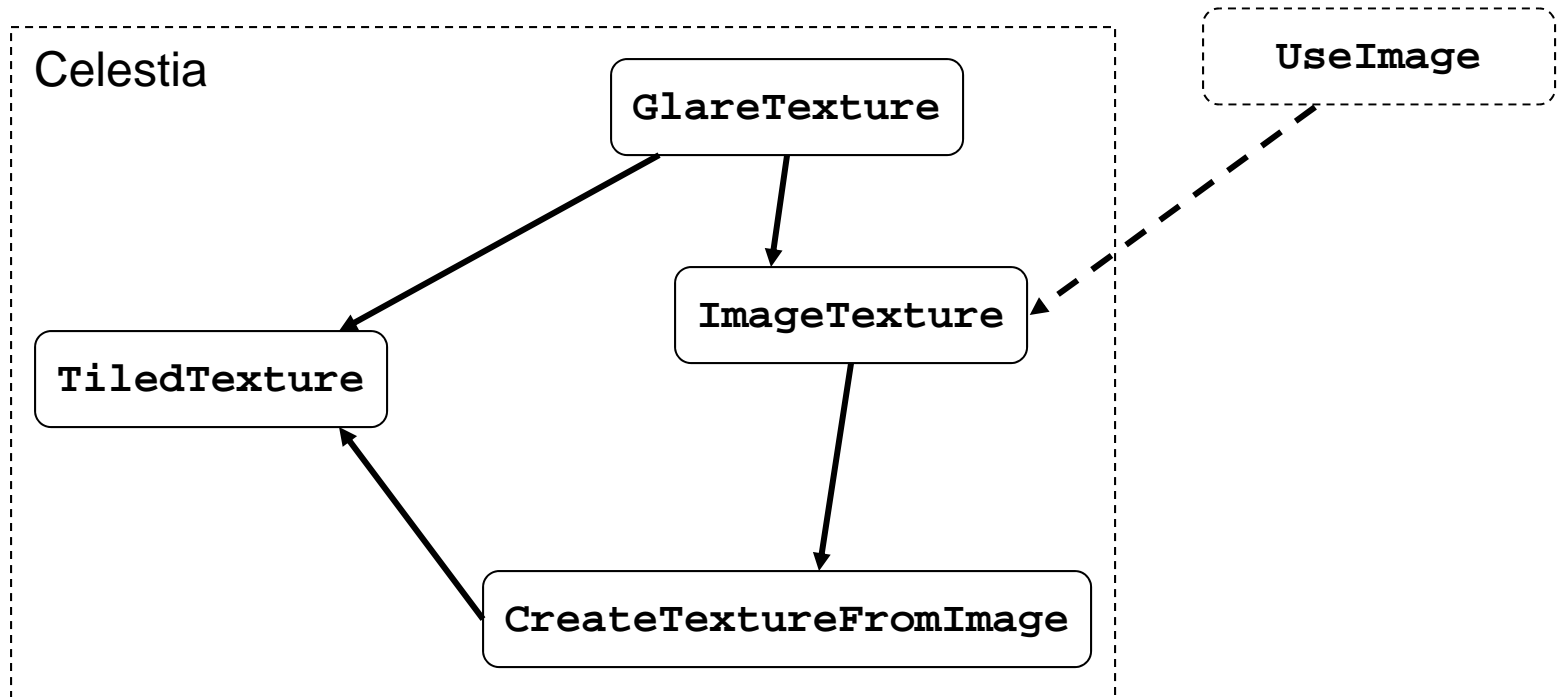
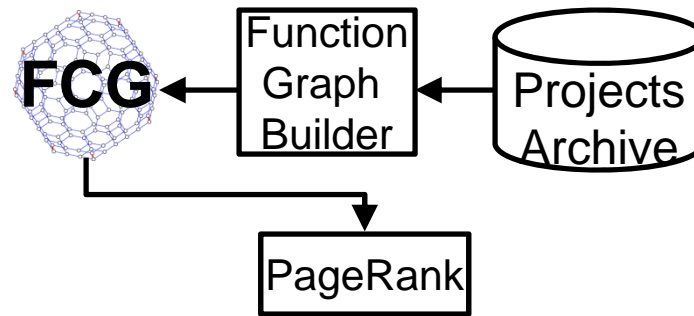




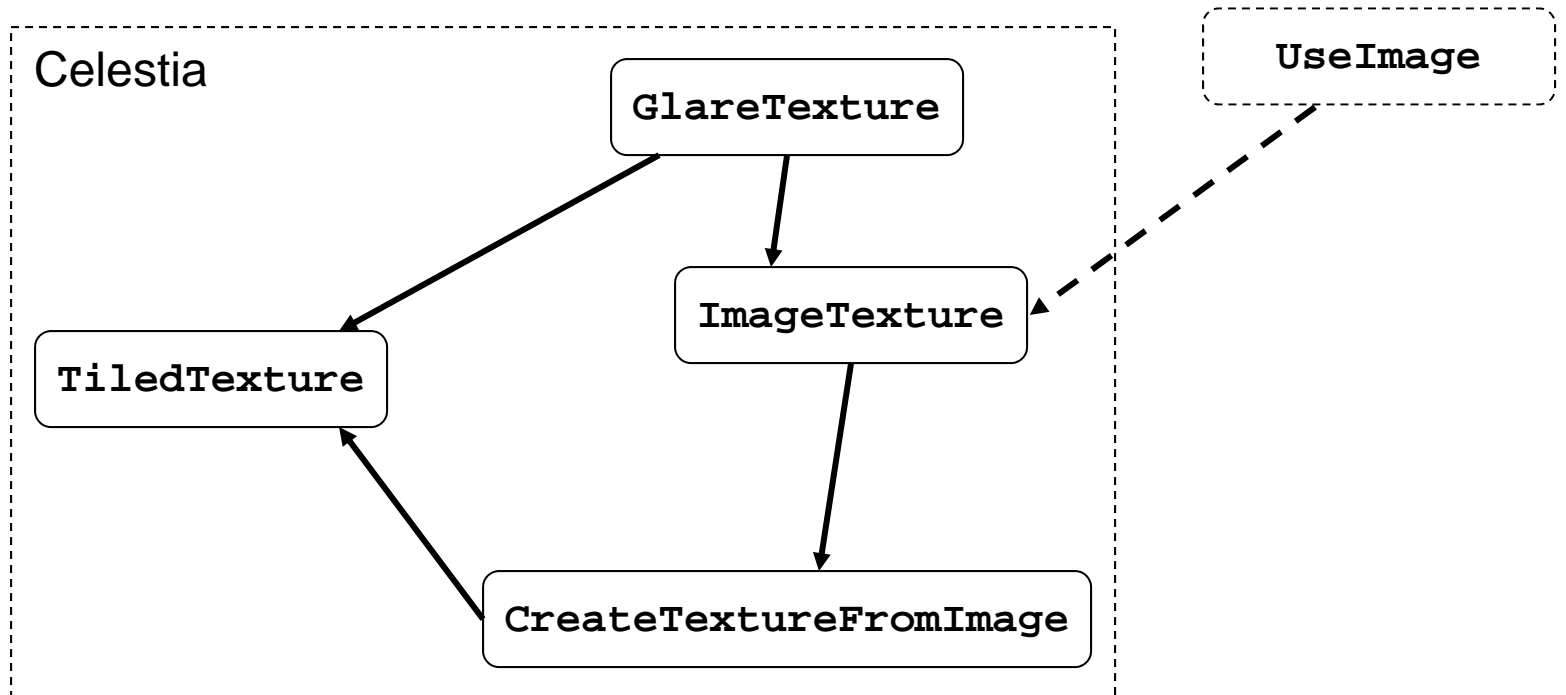
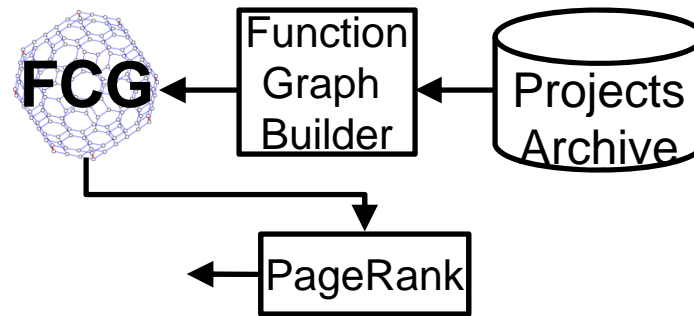


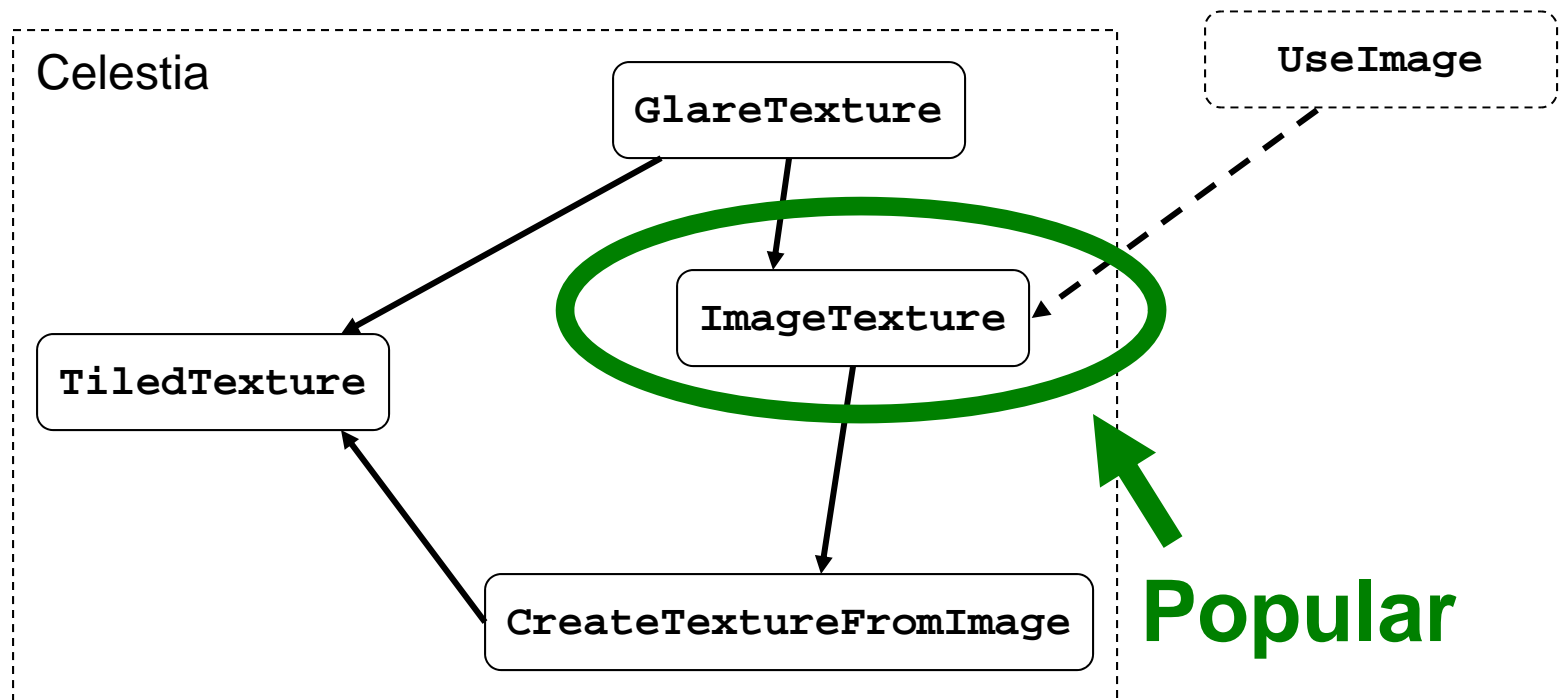
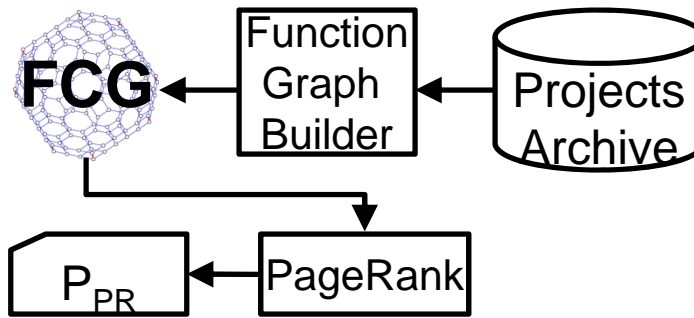


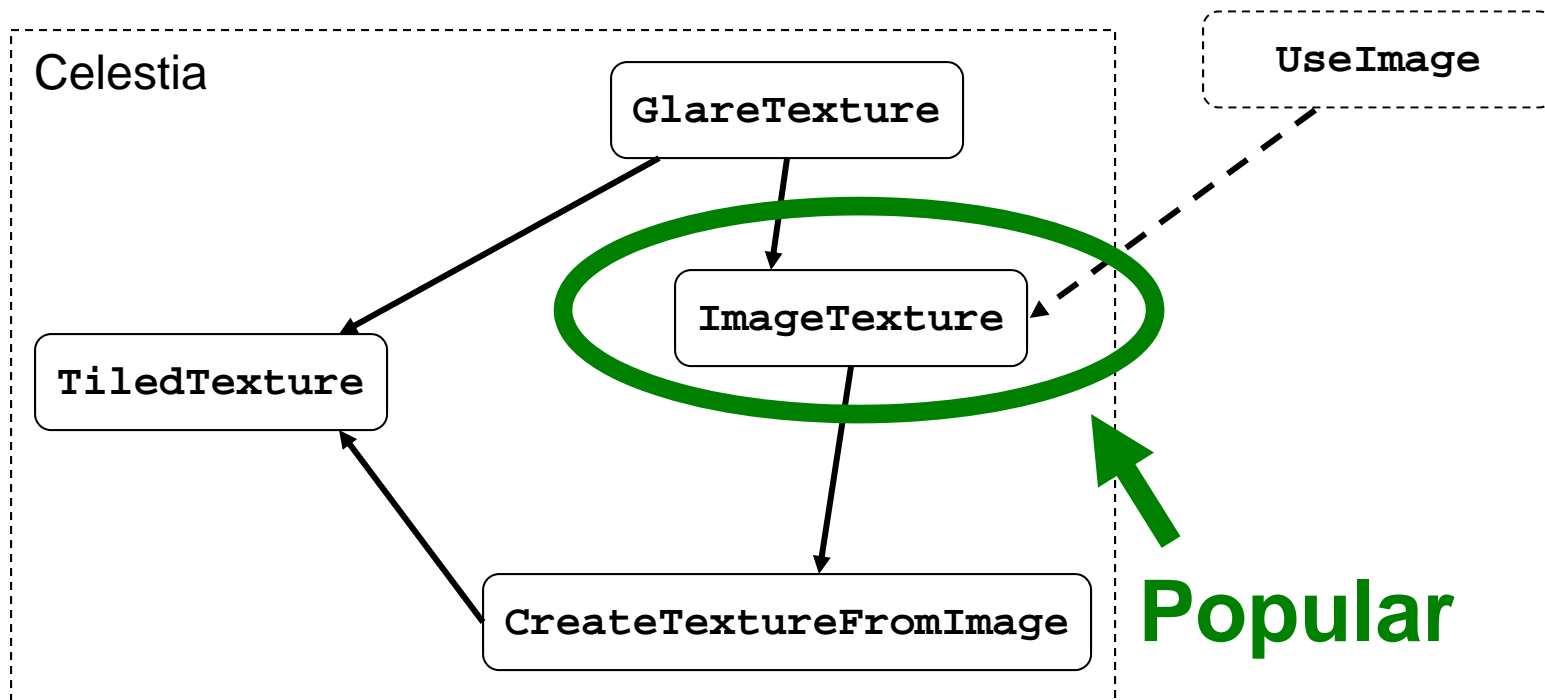
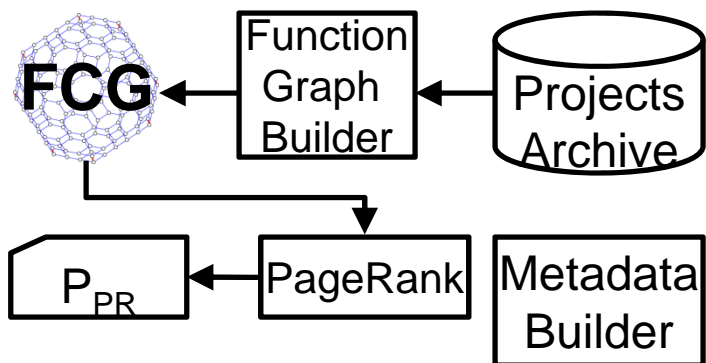


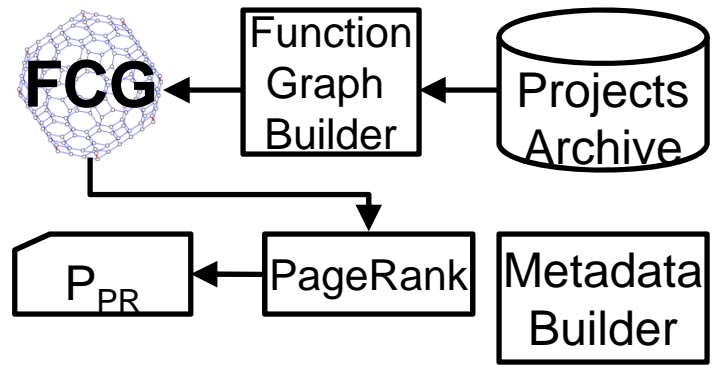


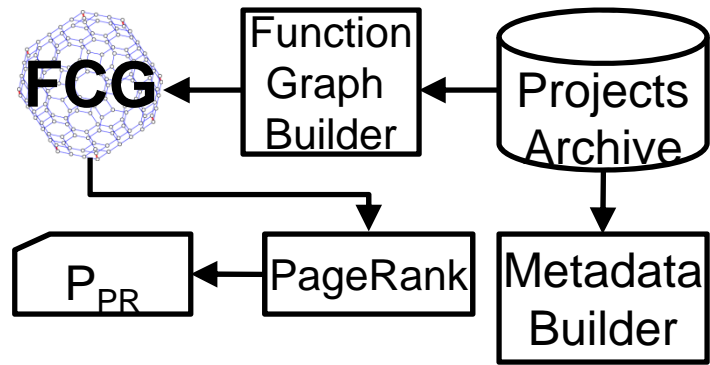


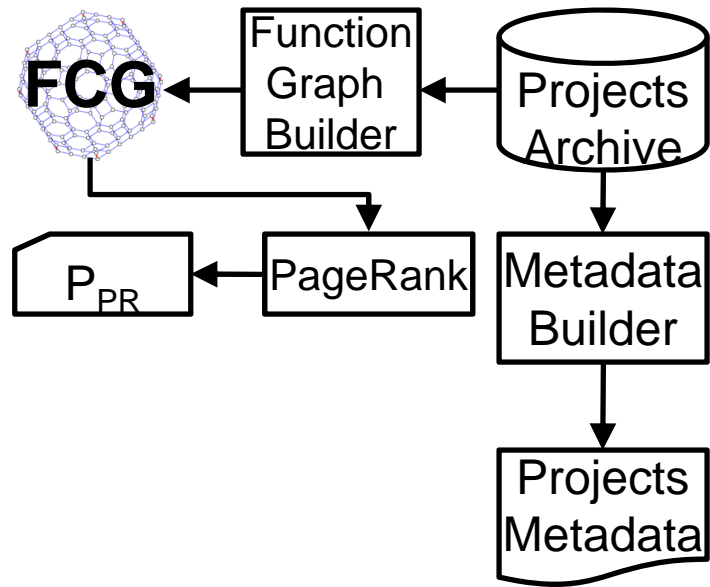


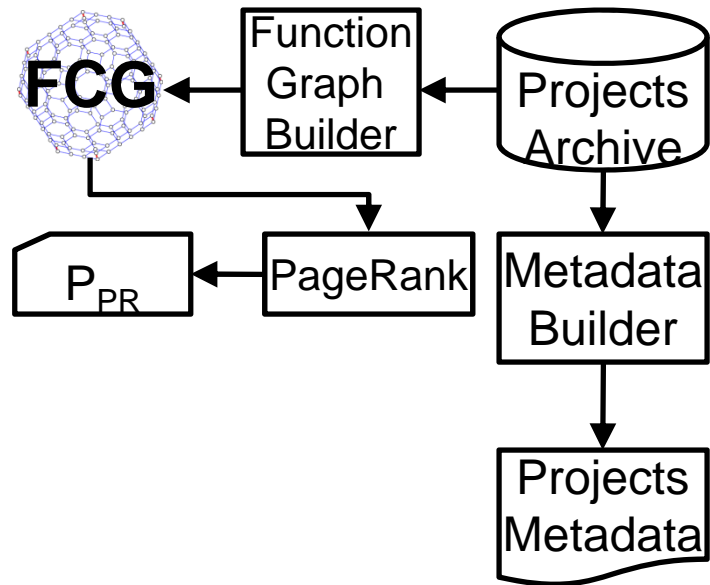




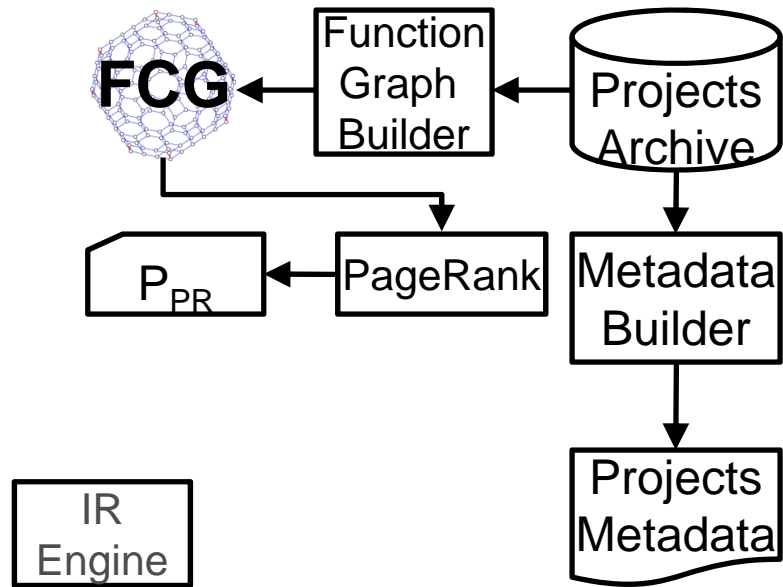








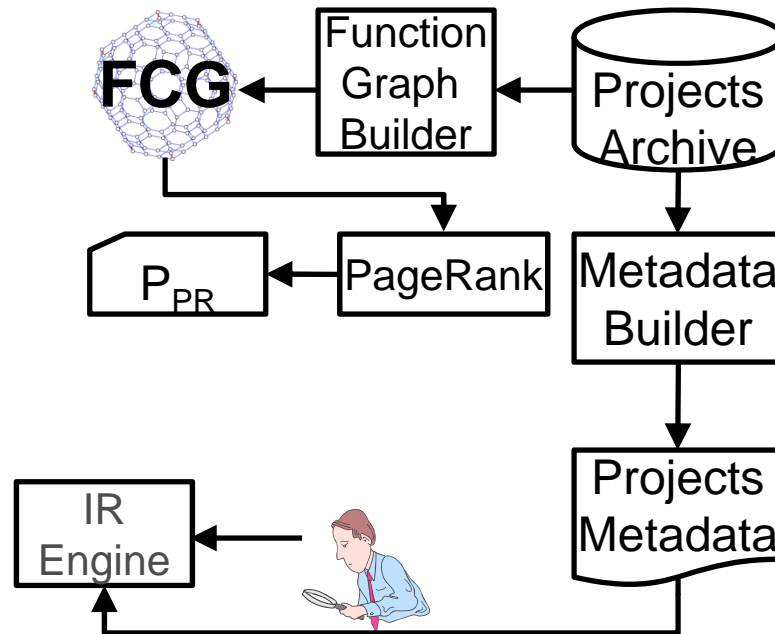
```
/* Tiling Method for Mip Maps */
static Texture* TiledTexture(Image& img)
{
    if (GetTextureCaps().nonPow2Supported)
    {
        /* prepares mip maps for dithering */
        if (mipMode == Texture::DefaultMipMaps)
            mipMode = Texture::AutoMipMaps;
    }
    ...
}
```



---

```
/* Tiling Method for Mip Maps */
static Texture* TiledTexture(Image& img)
{
    if (GetTextureCaps().nonPow2Supported)
    {
        /* prepares mip maps for dithering */
        if (mipMode == Texture::DefaultMipMaps)
            mipMode = Texture::AutoMipMaps;
    }
    ...
}
```



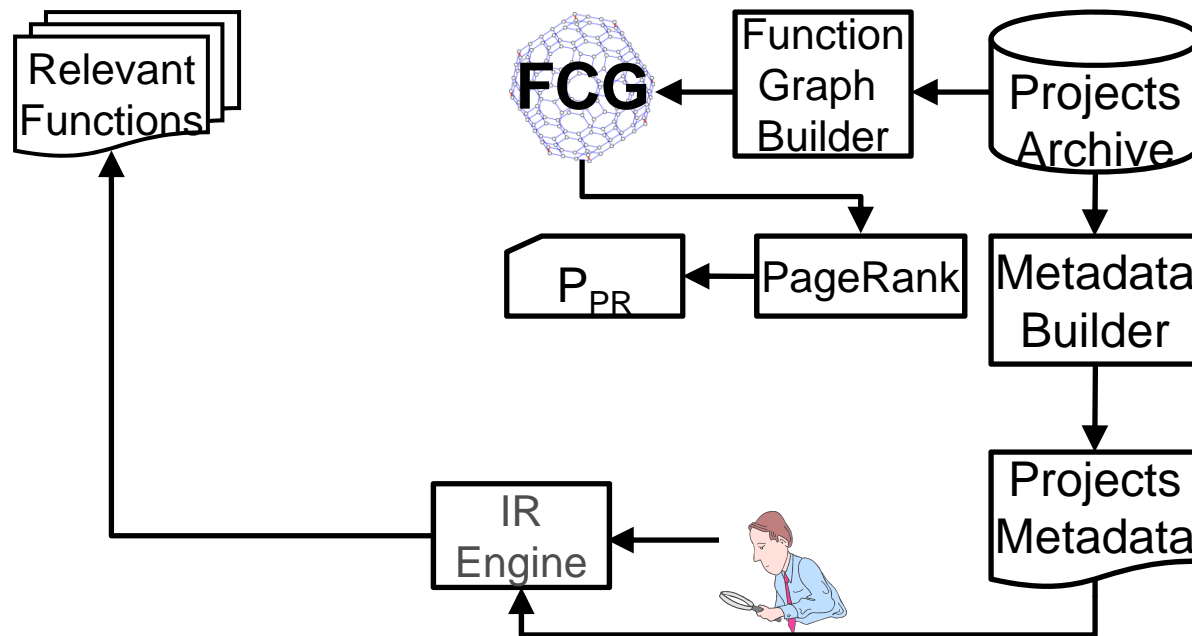


“mip map dithering” →

```

/* Tiling Method for Mip Maps */
static Texture* TiledTexture(Image& img)
{
    if (GetTextureCaps().nonPow2Supported)
    {
        /* prepares mip maps for dithering */
        if (mipMode == Texture::DefaultMipMaps)
            mipMode = Texture::AutoMipMaps;
    }
    ...
}

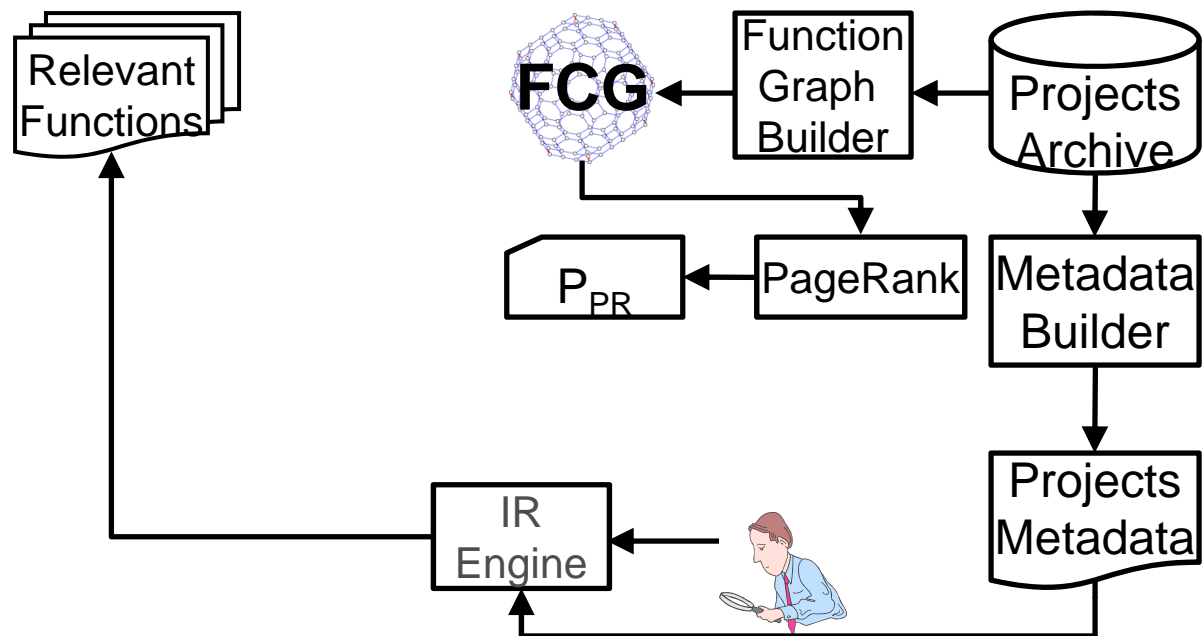
```

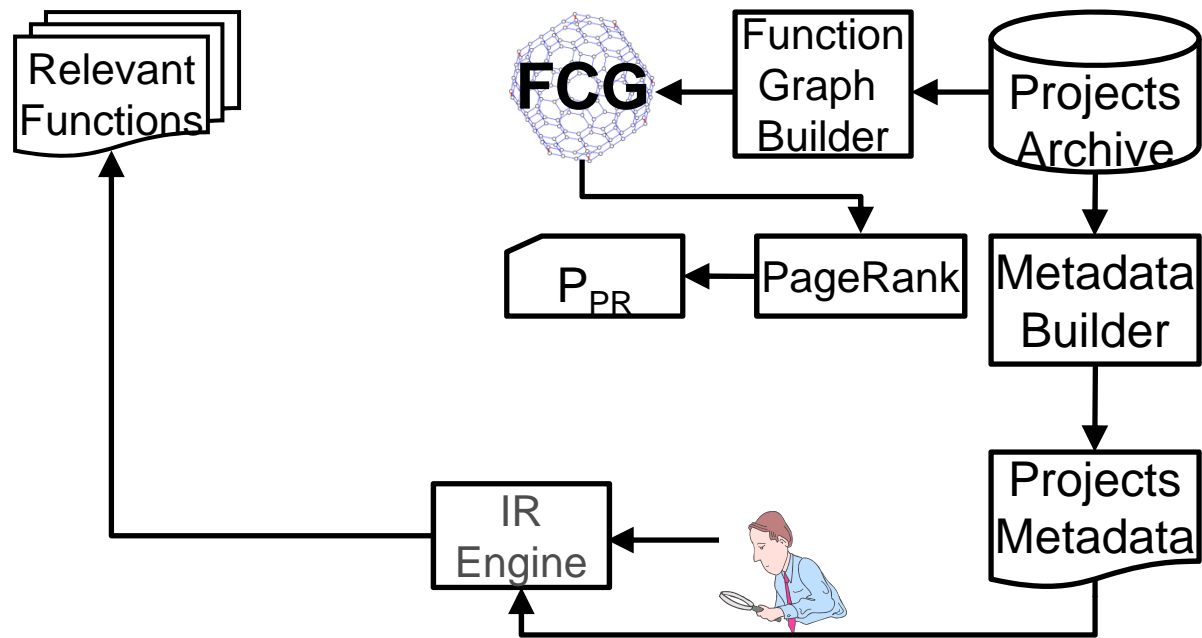


“mip map dithering” →

```

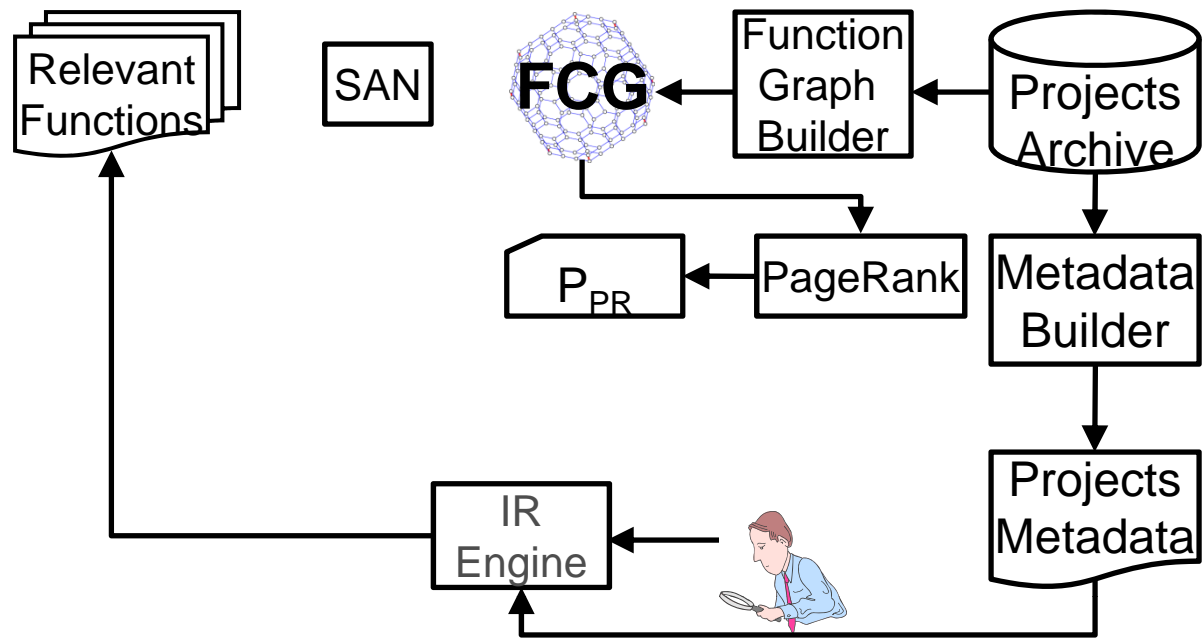
/* Tiling Method for Mip Maps */
static Texture* TiledTexture(Image& img)
{
    if (GetTextureCaps().nonPow2Supported)
    {
        /* prepares mip maps for dithering */
        if (mipMode == Texture::DefaultMipMaps)
            mipMode = Texture::AutoMipMaps;
    }
    ...
}
  
```





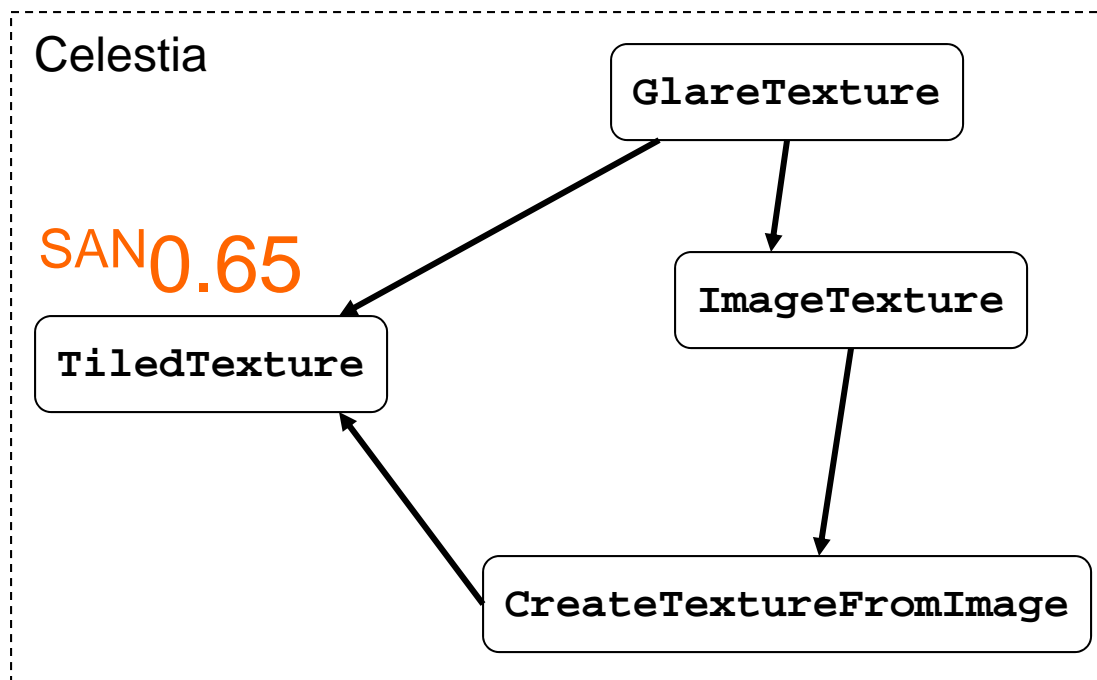
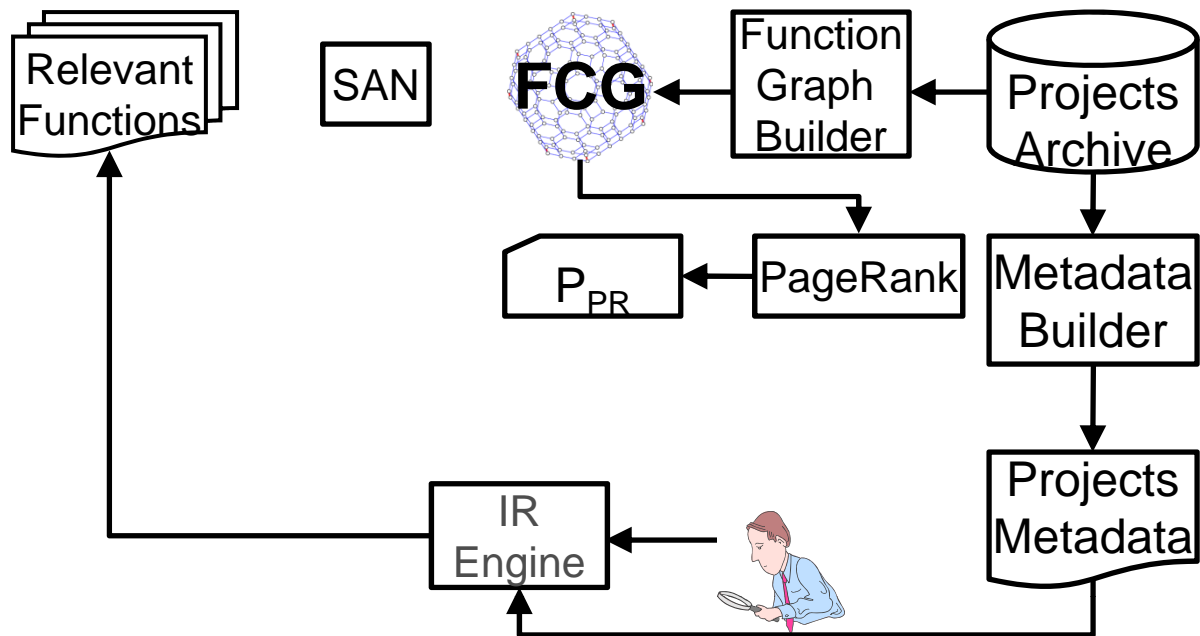
SAN0.65

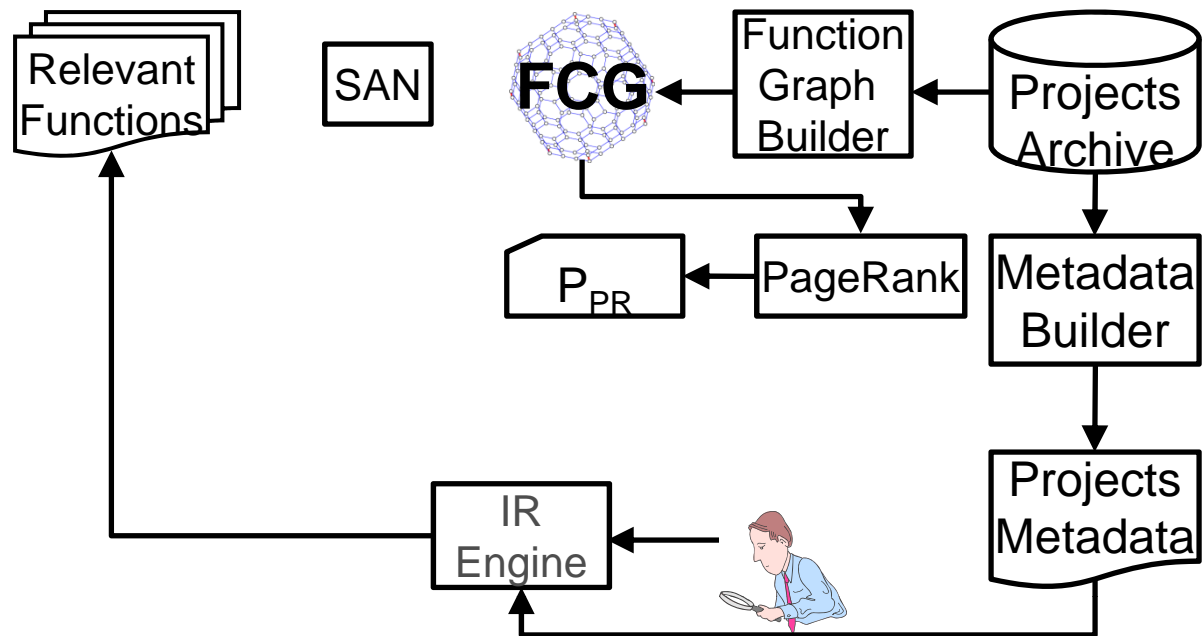
TiledTexture



SAN0.65

TiledTexture



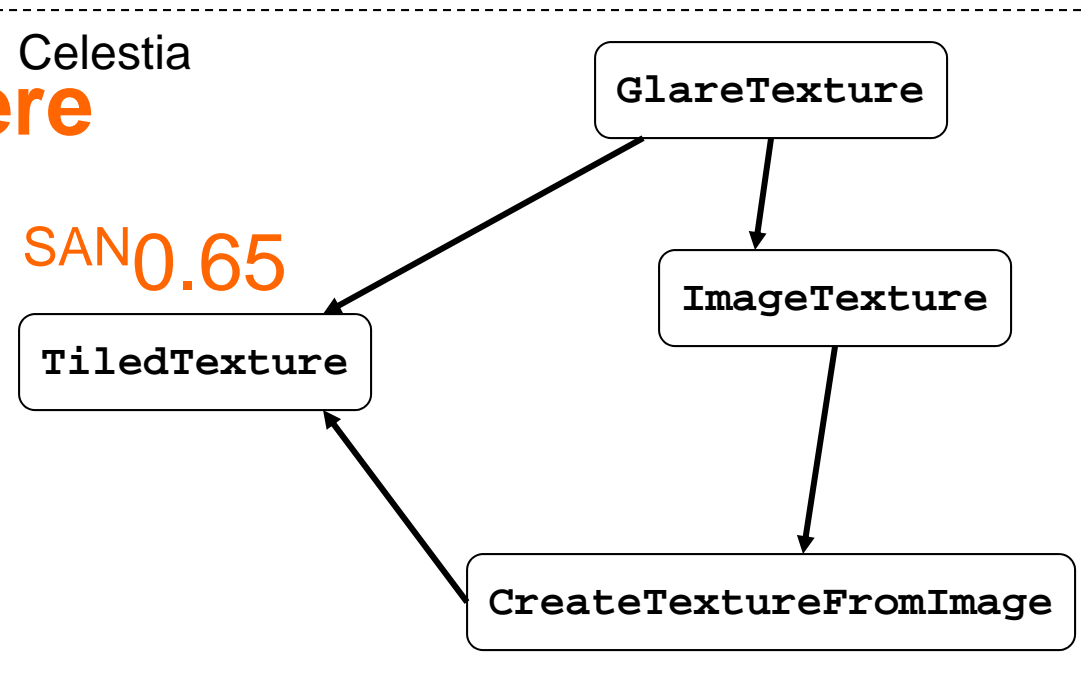


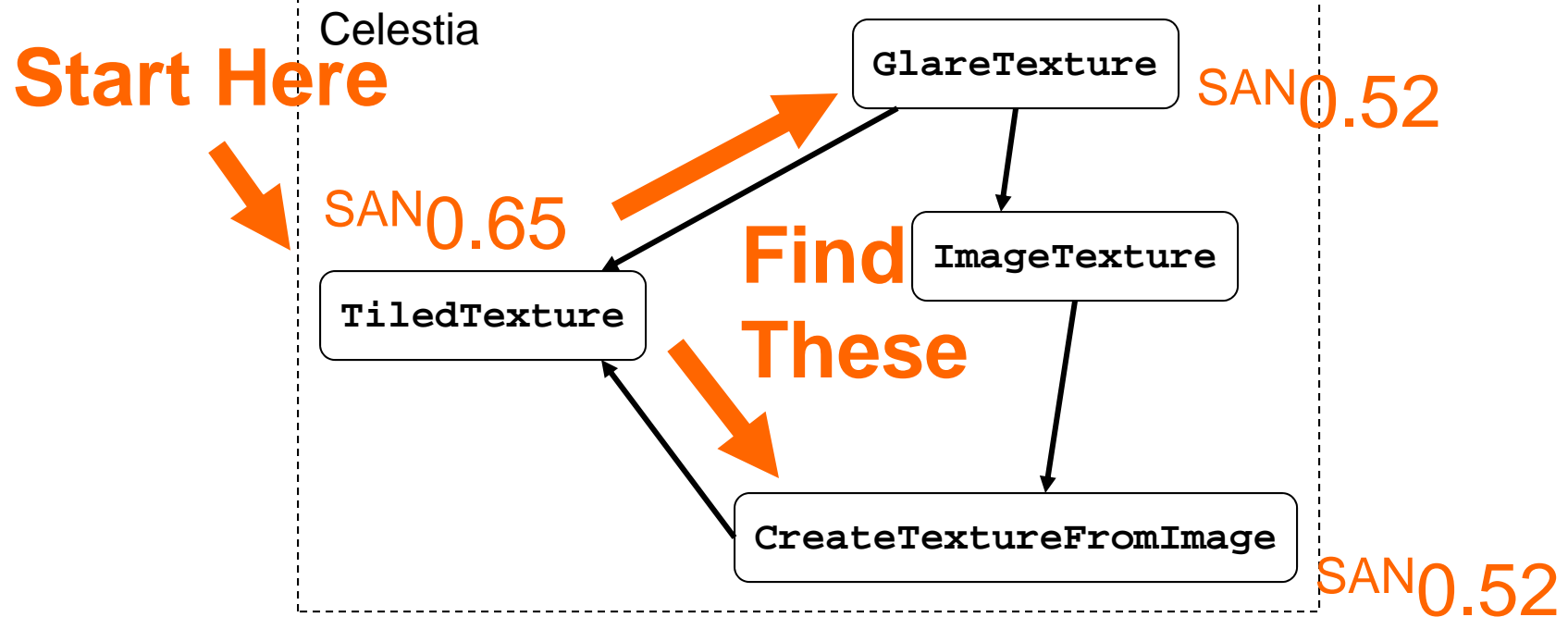
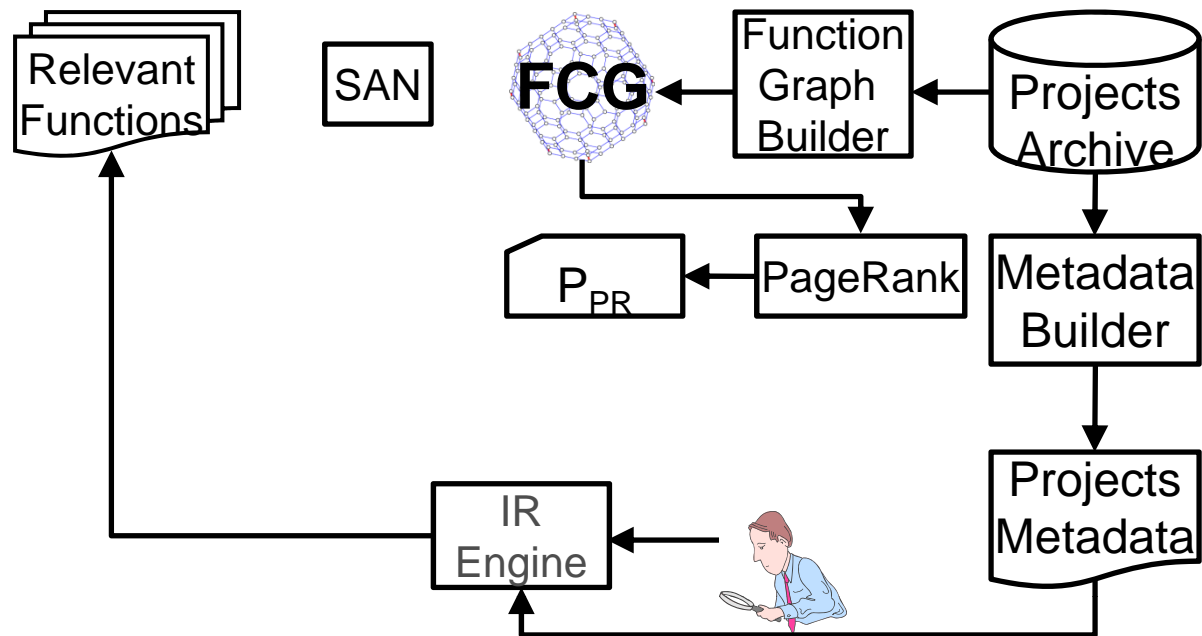
**Start Here**



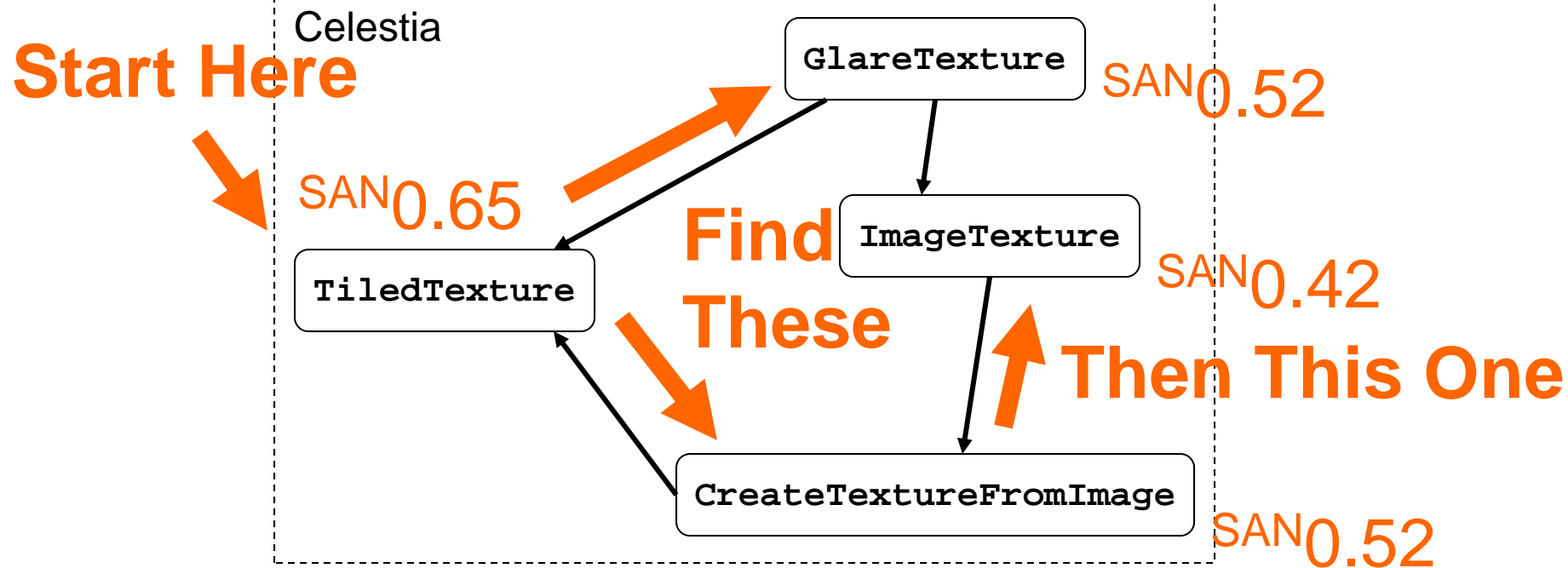
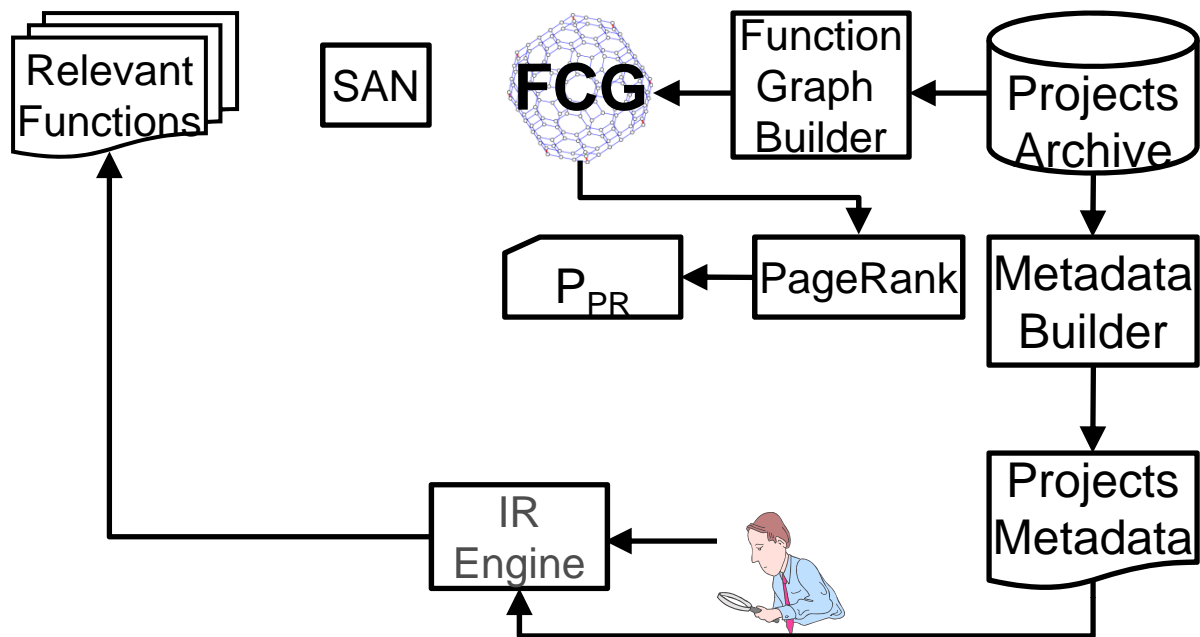
Celestia

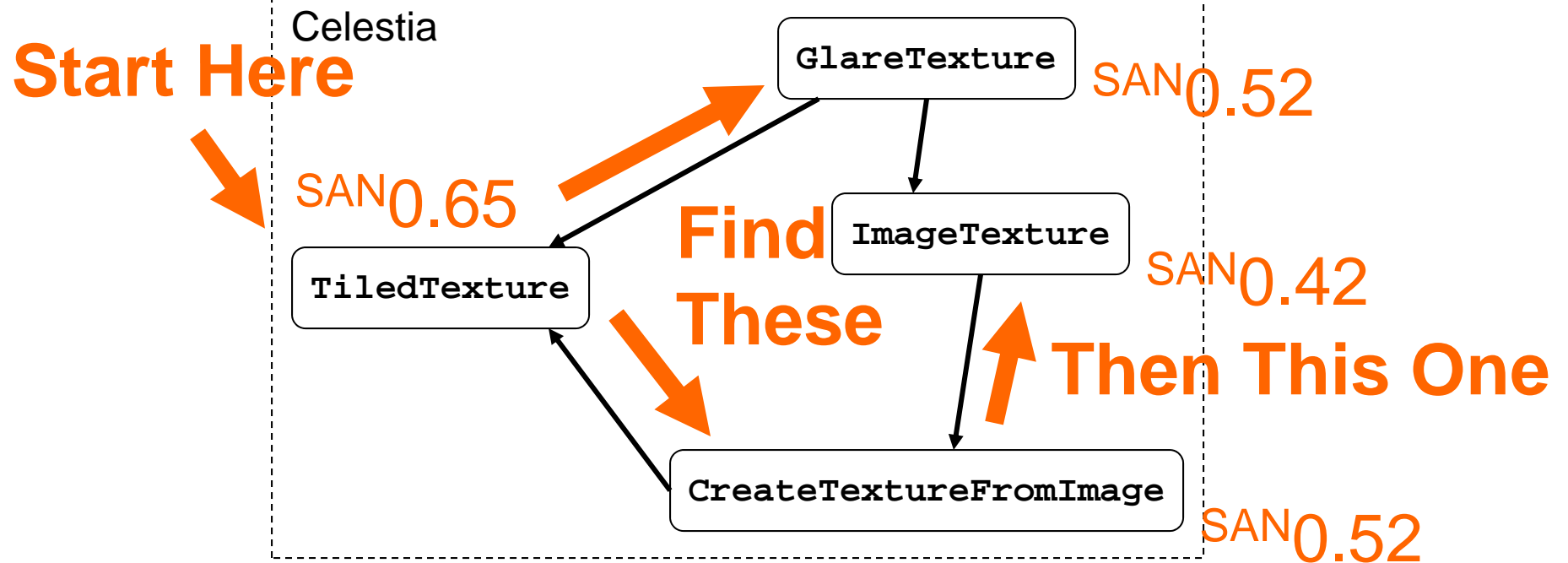
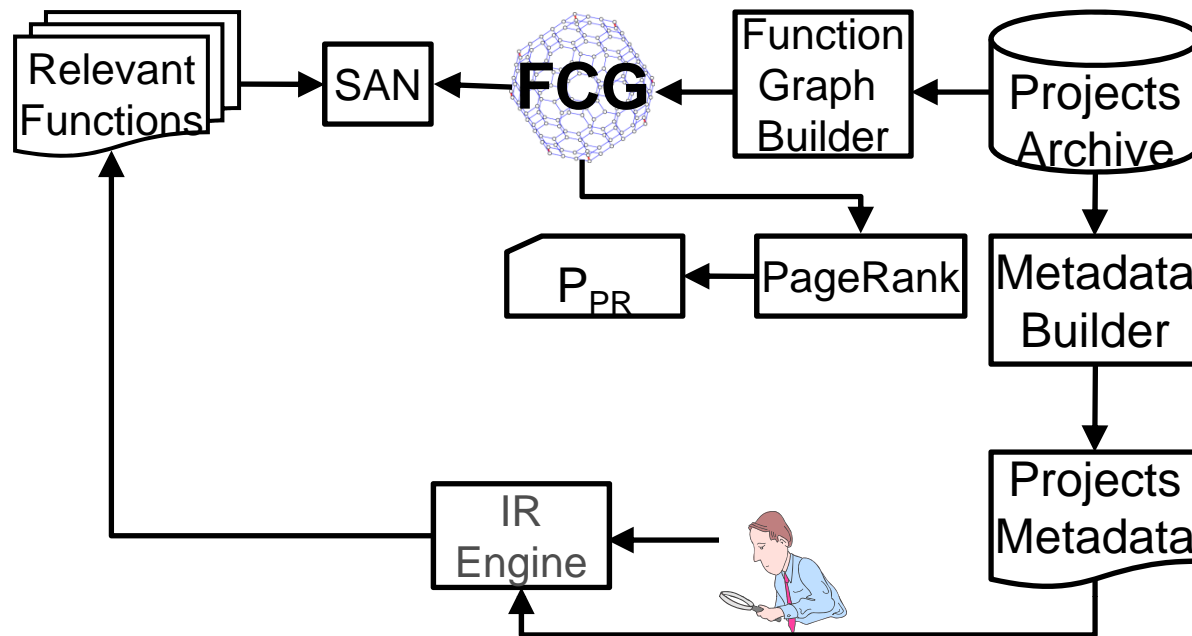
SAN 0.65

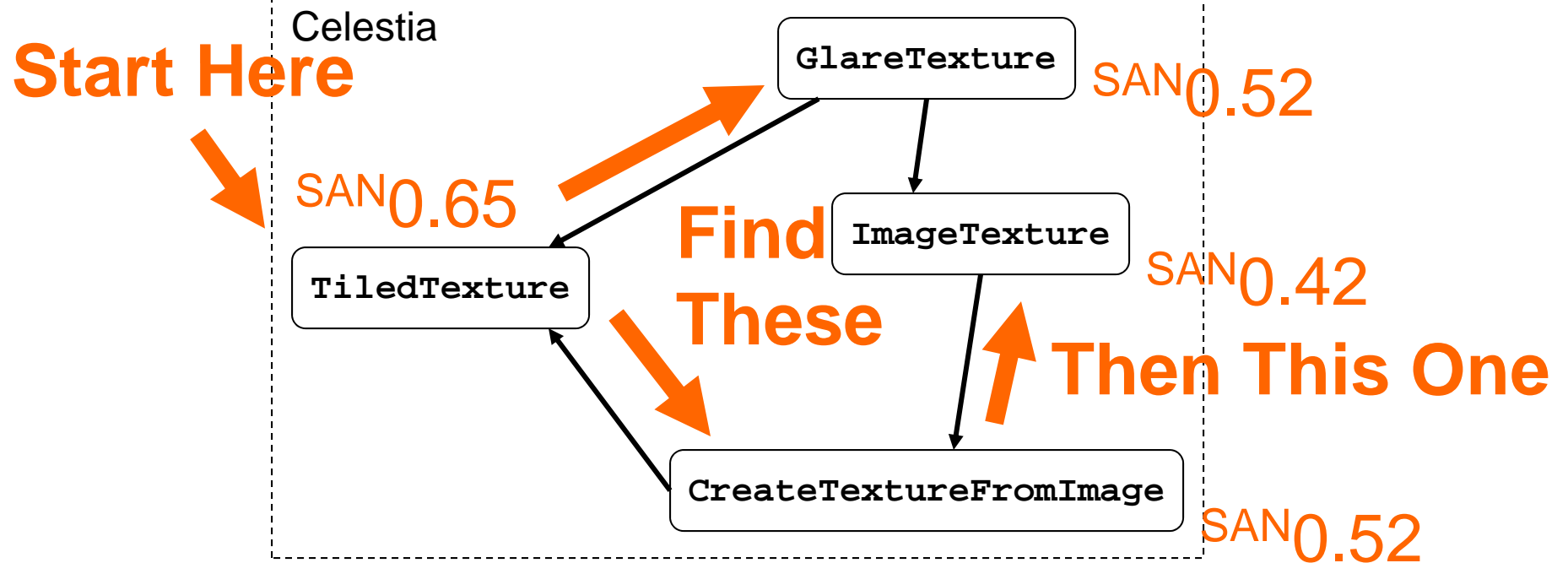
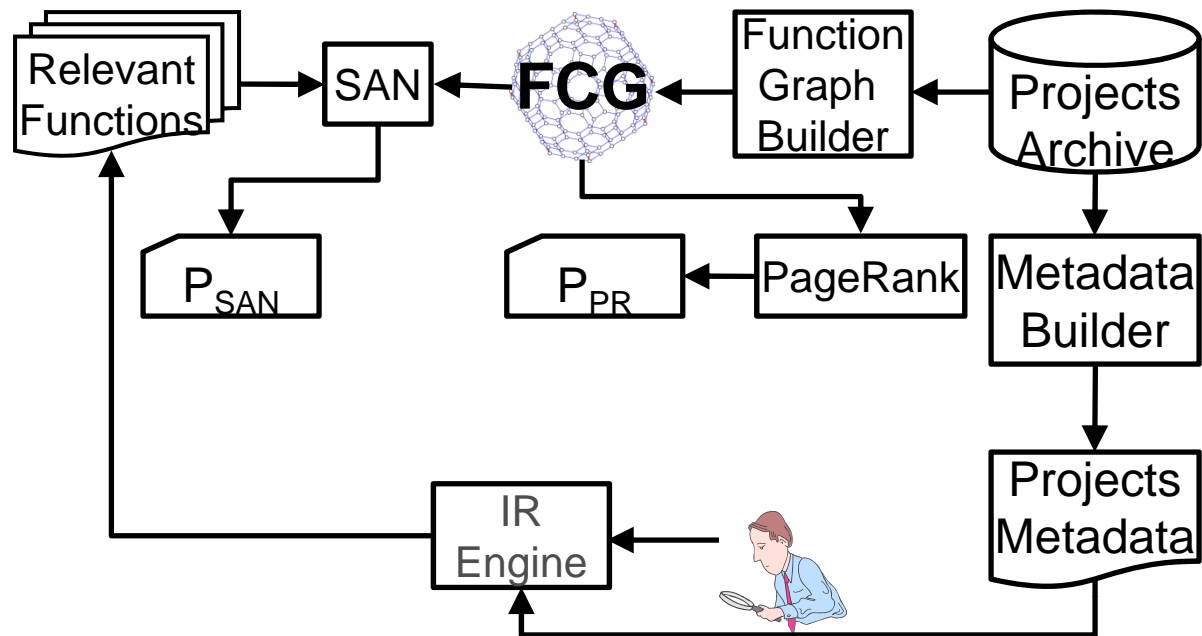


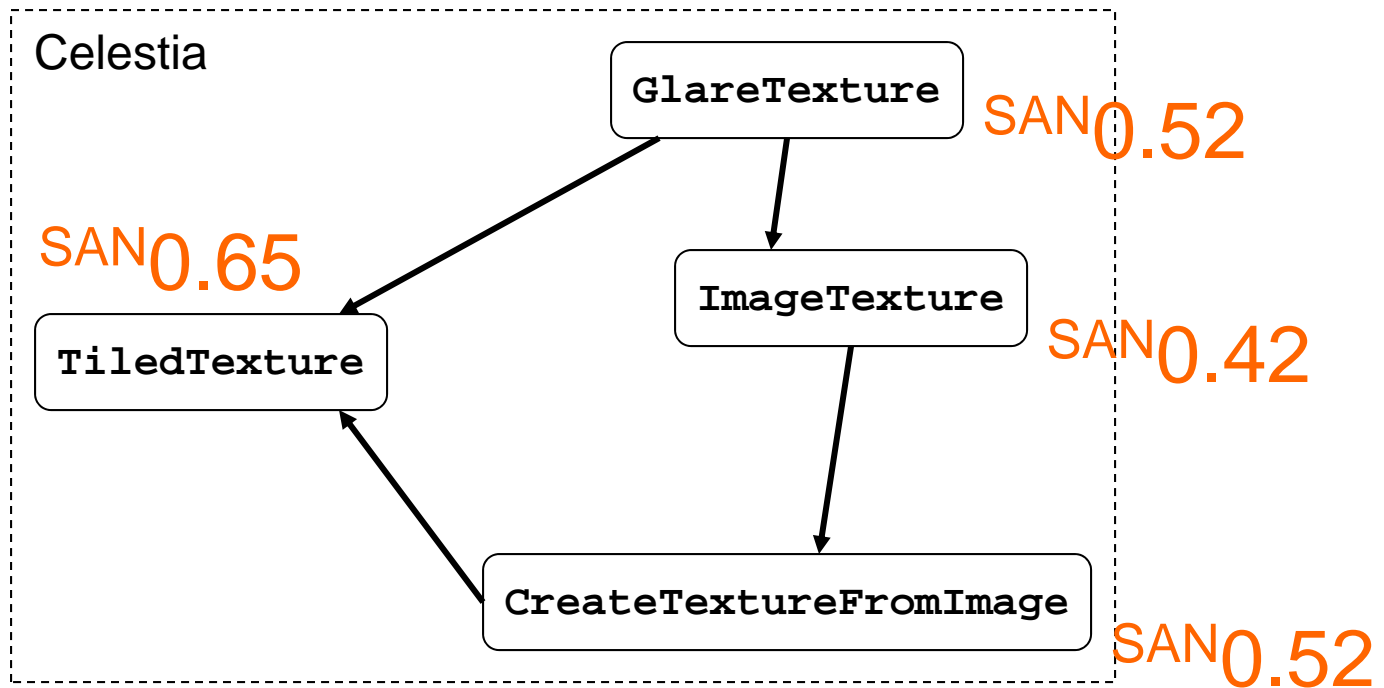
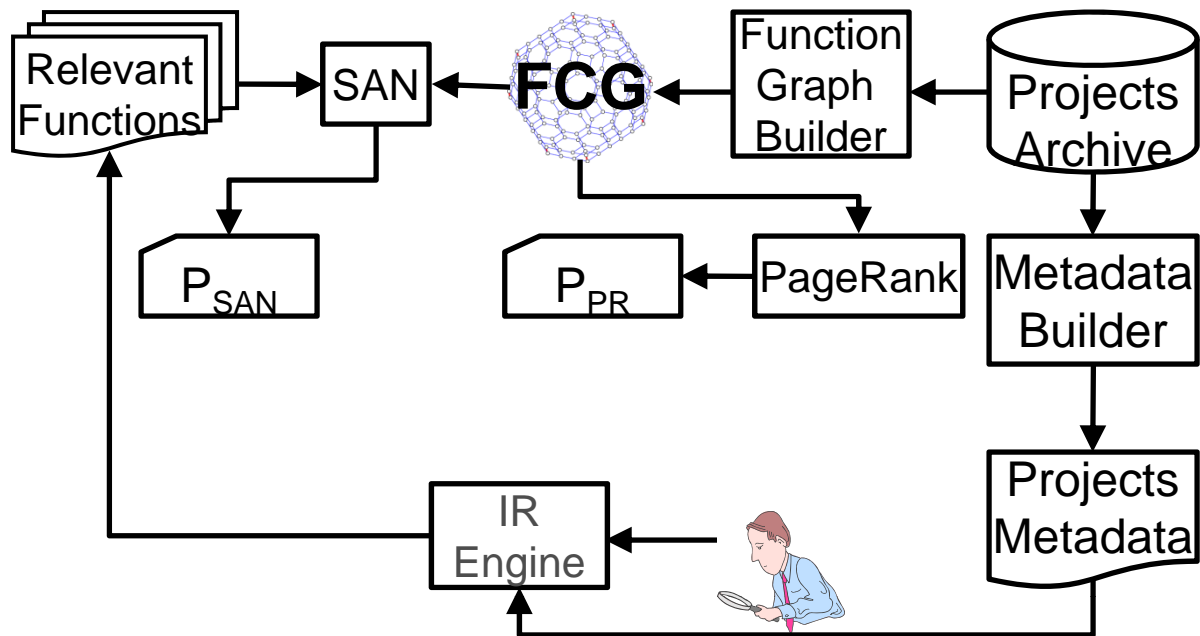


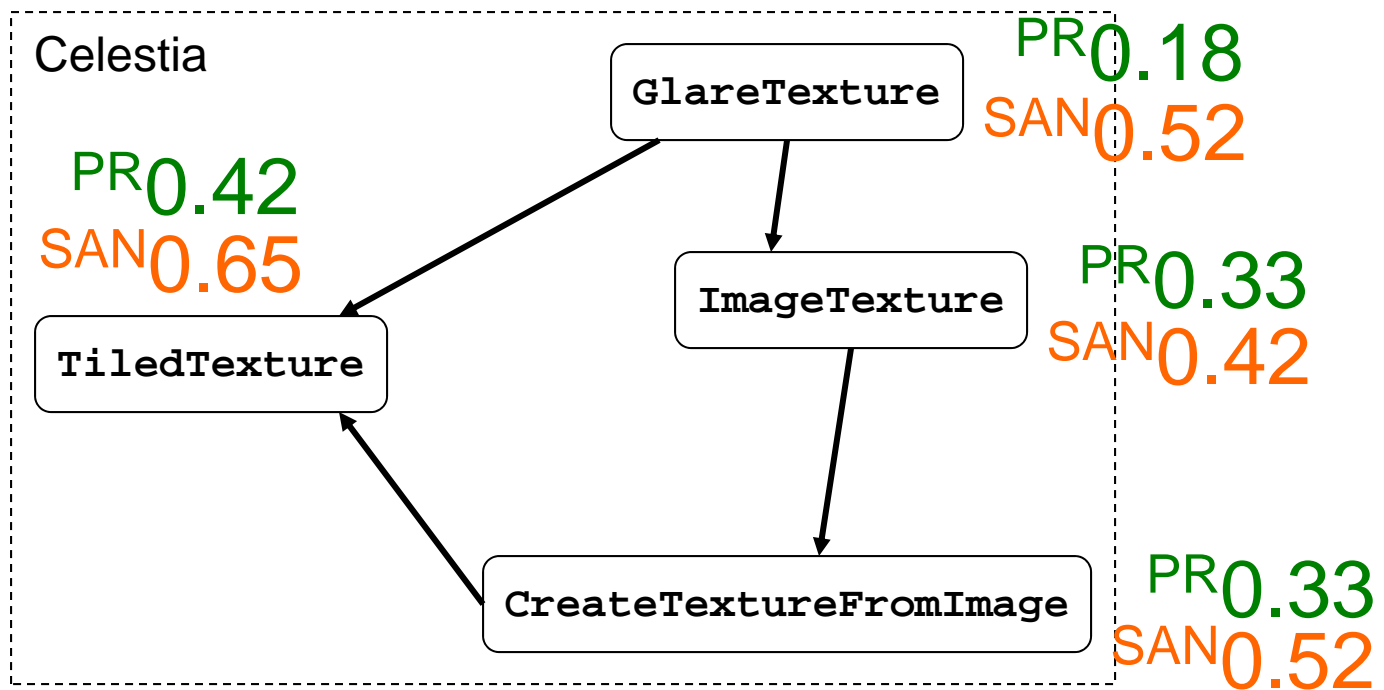
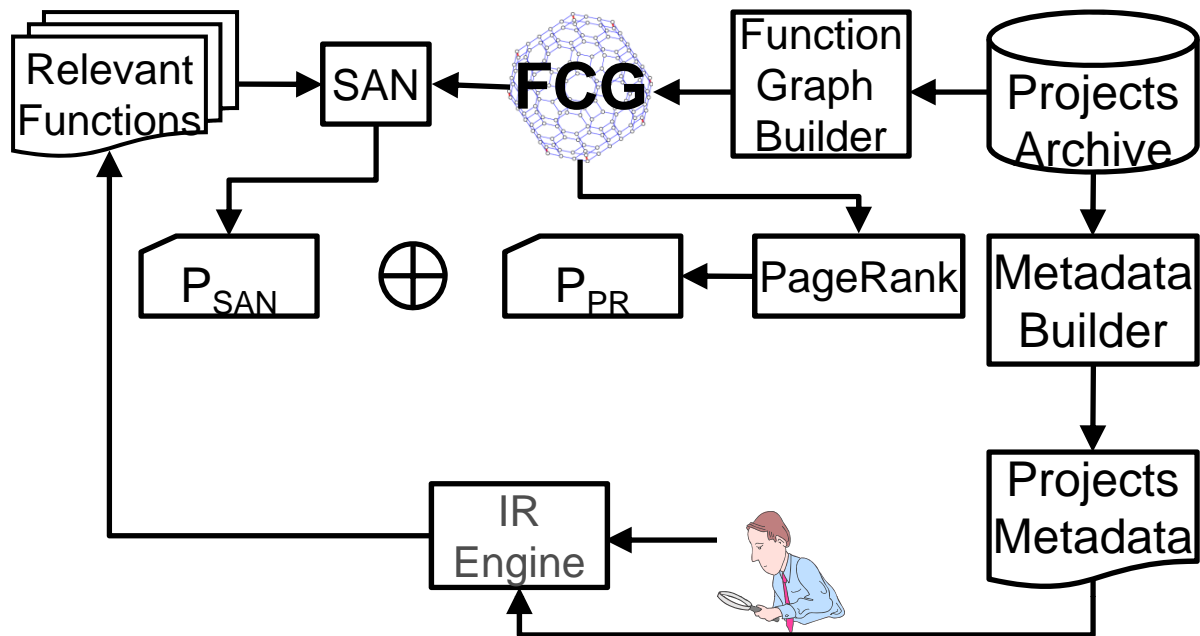


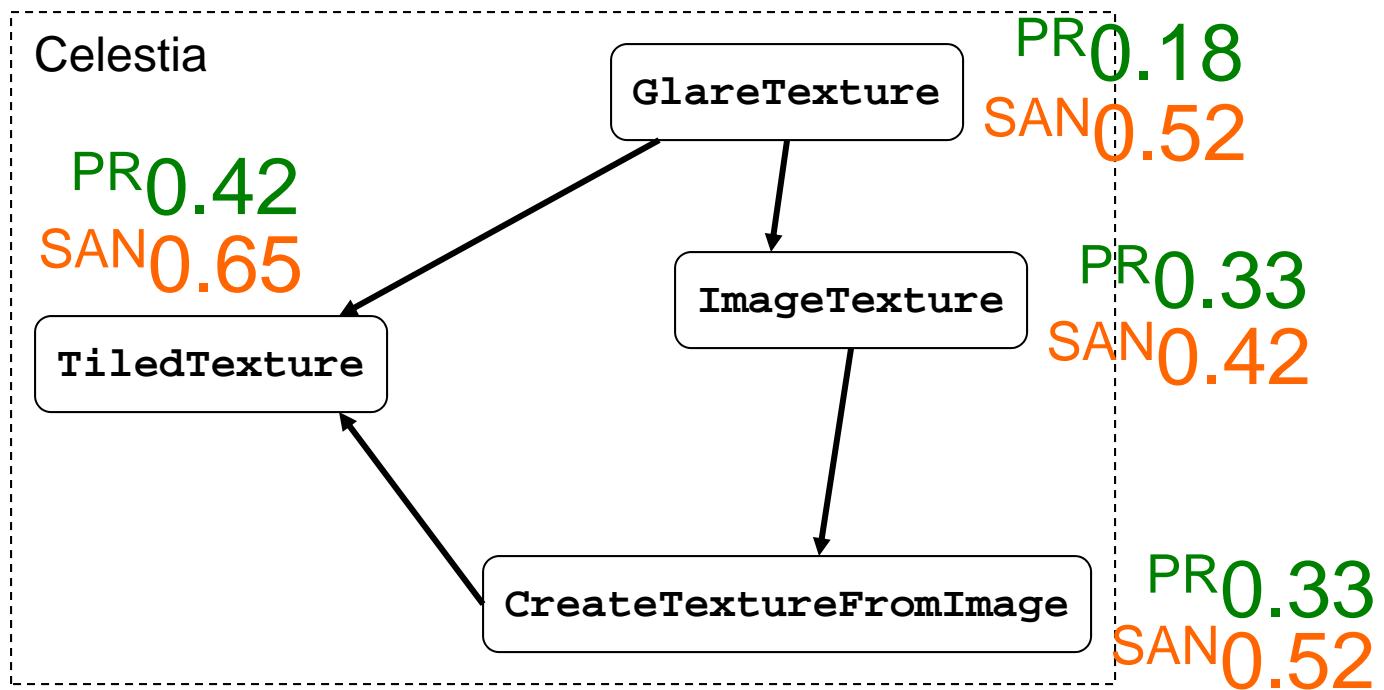
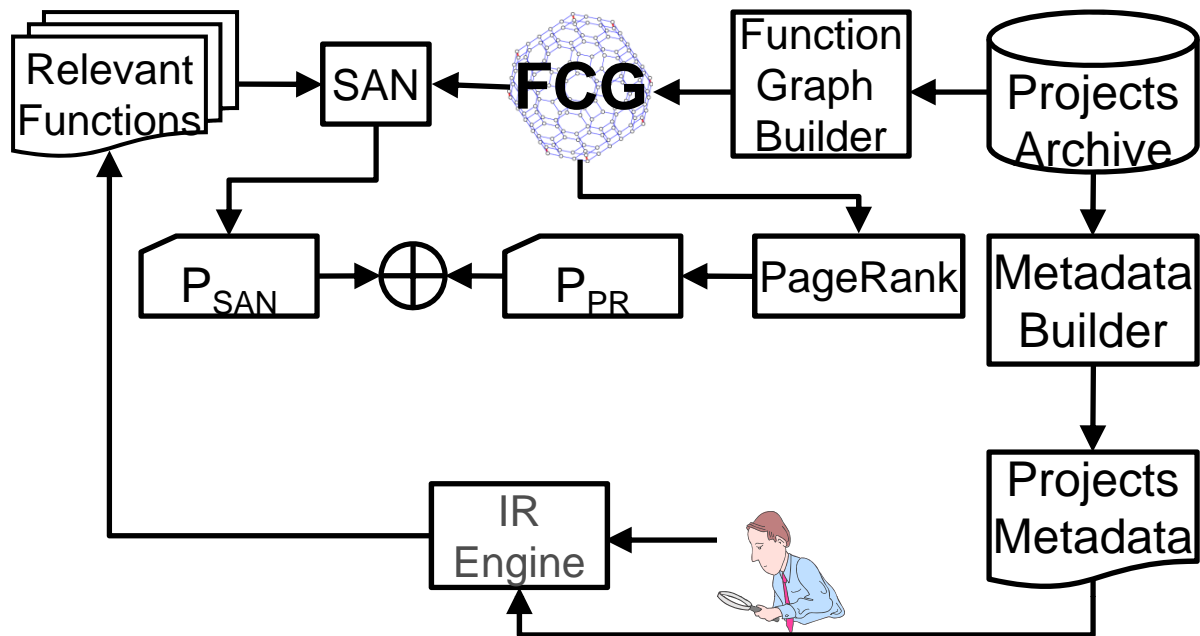


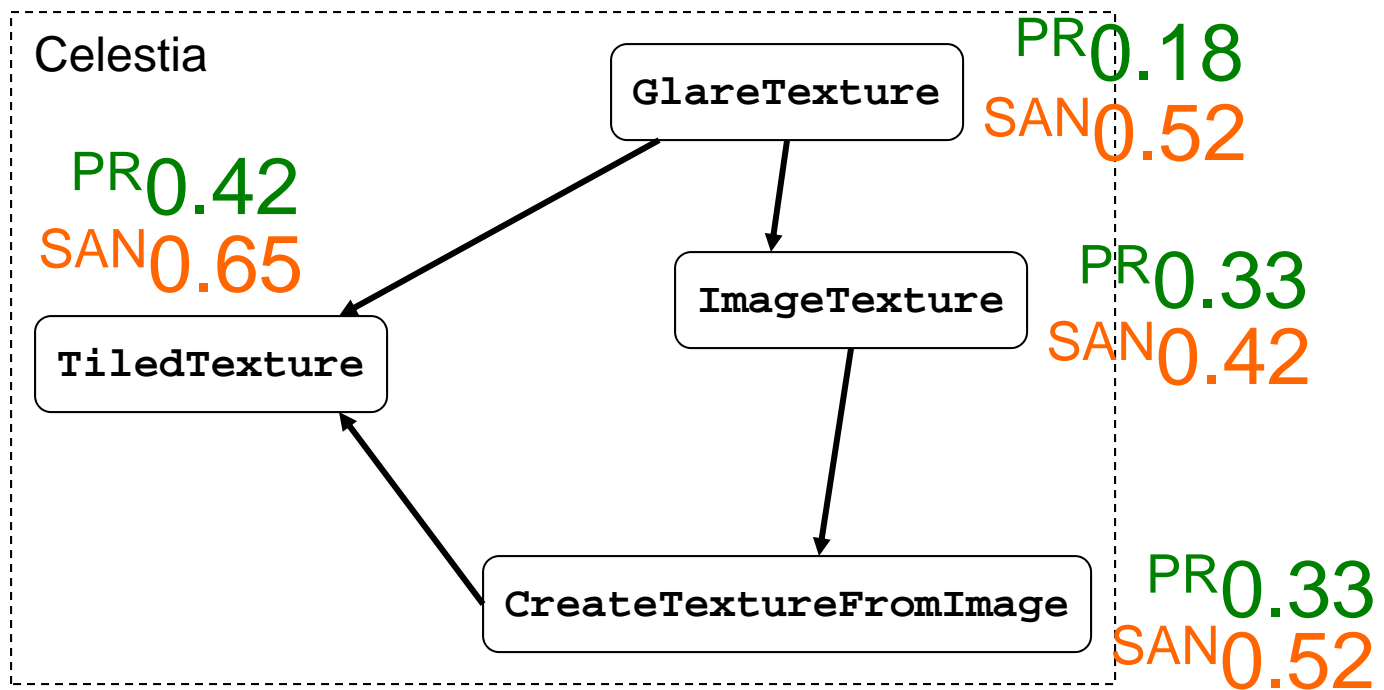
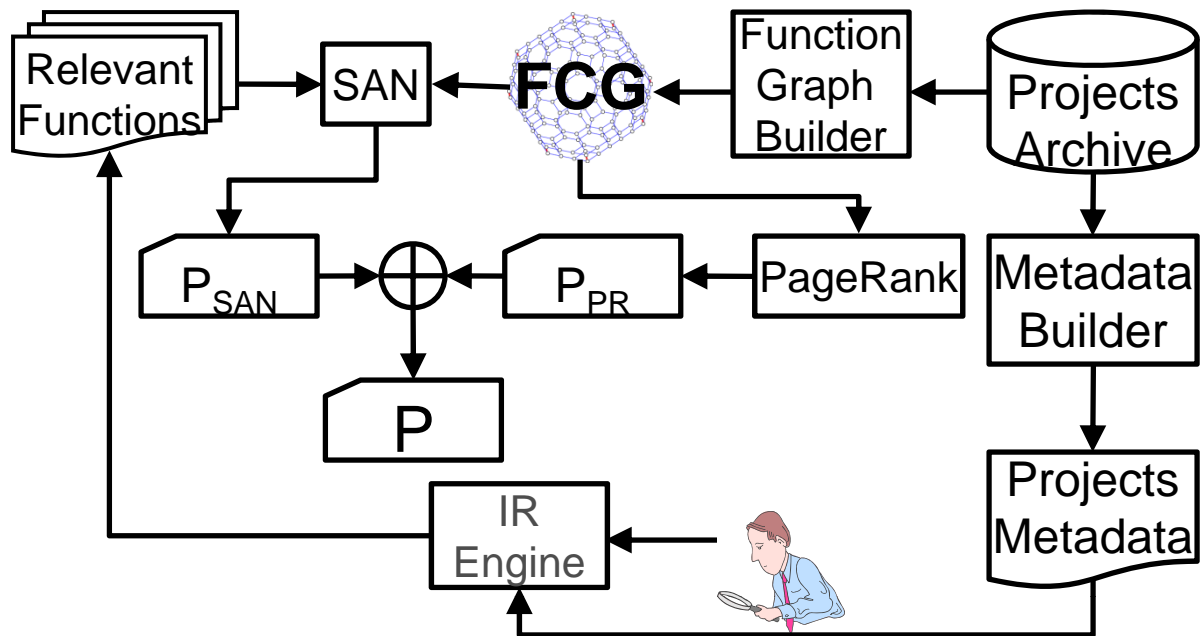


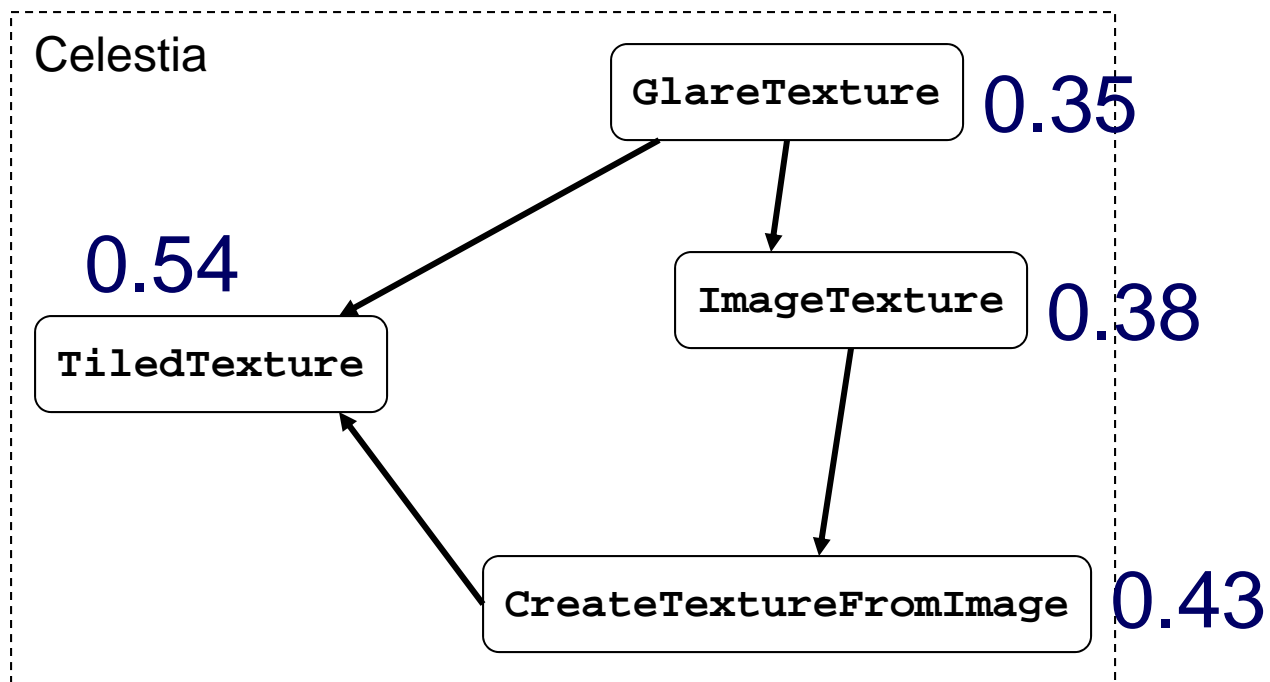
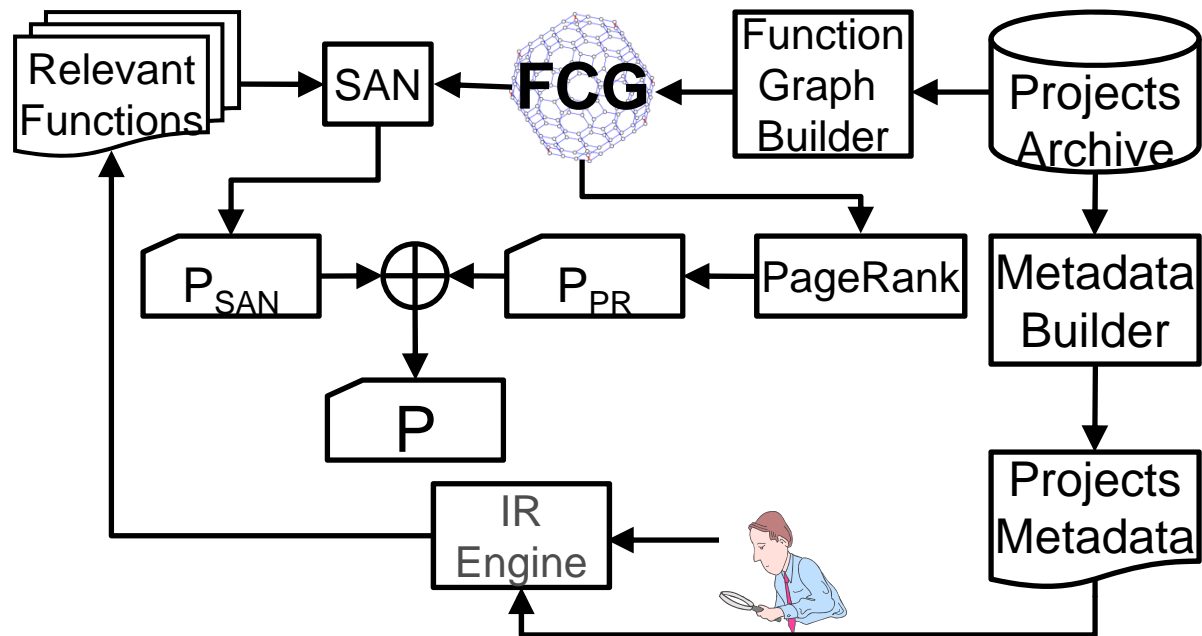




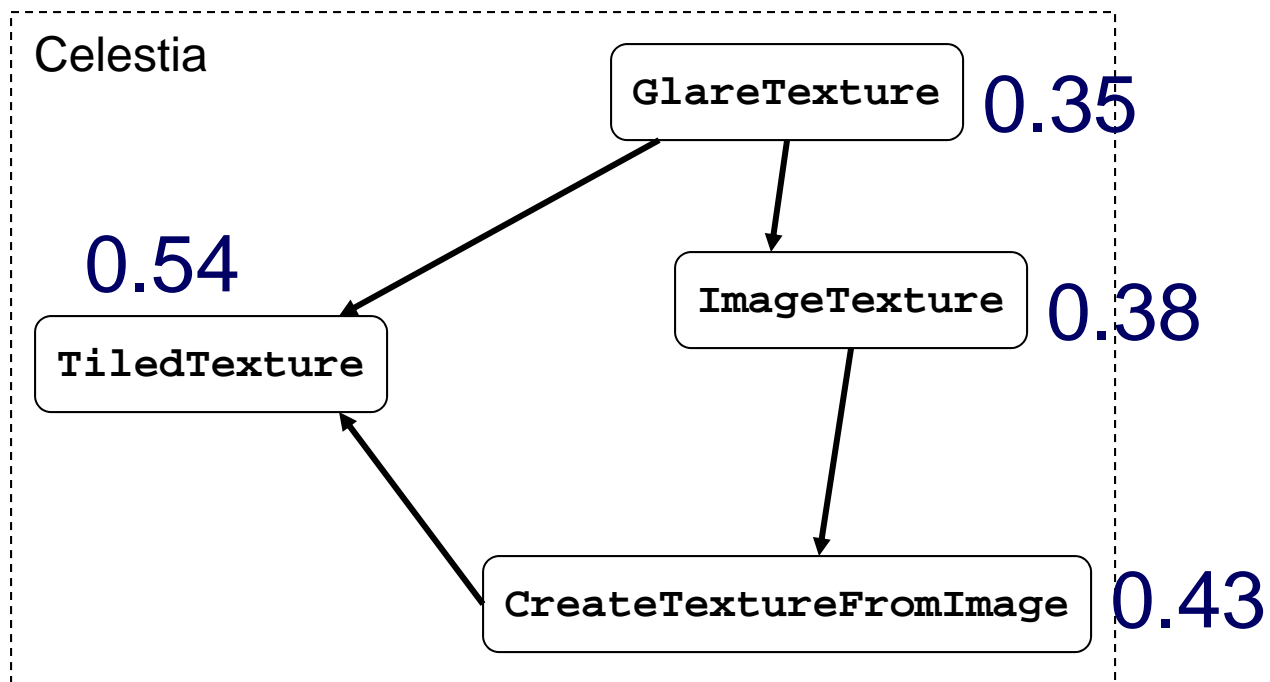
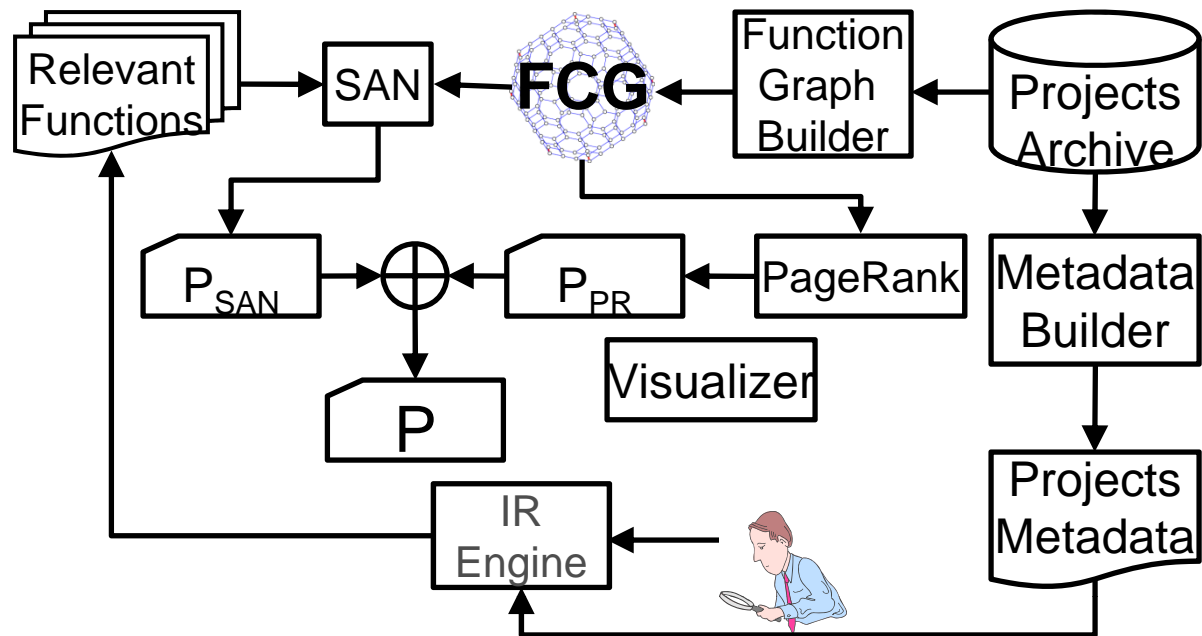


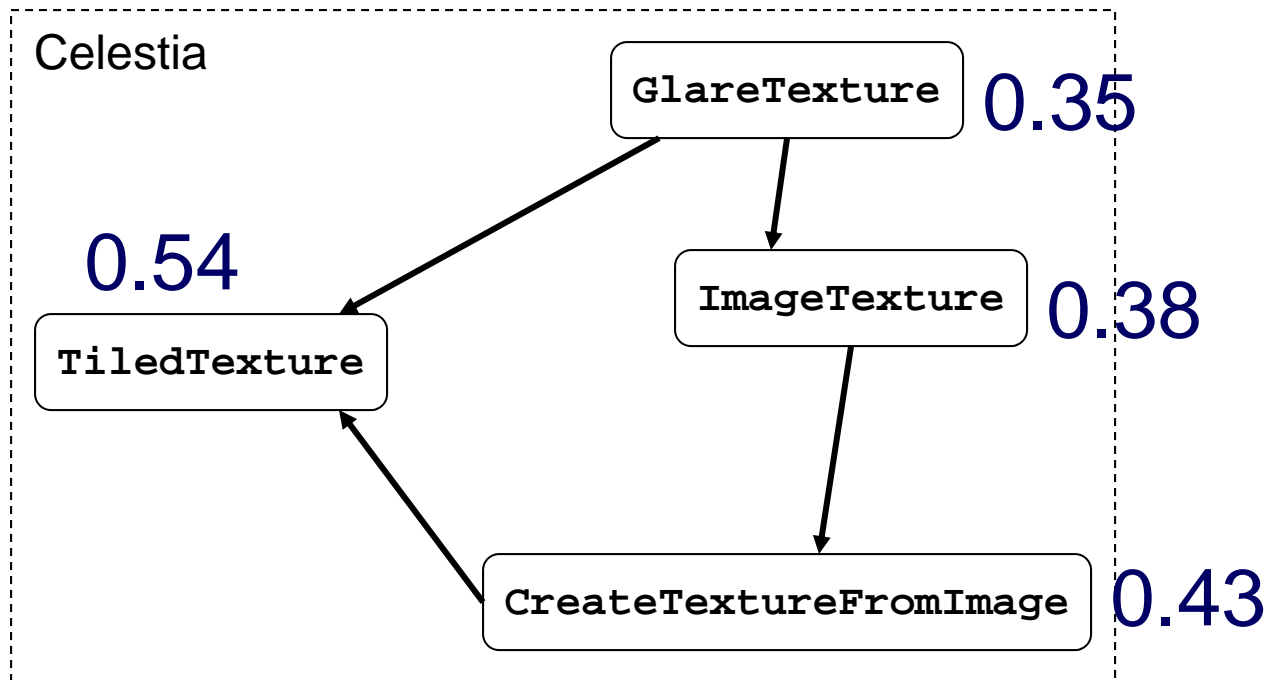
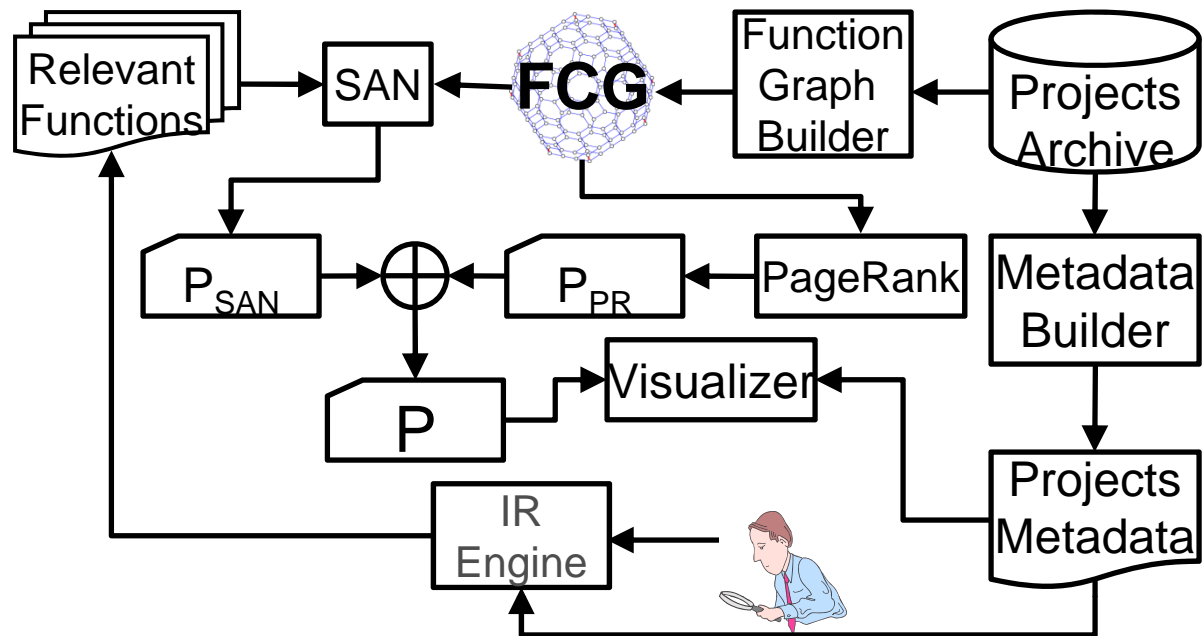


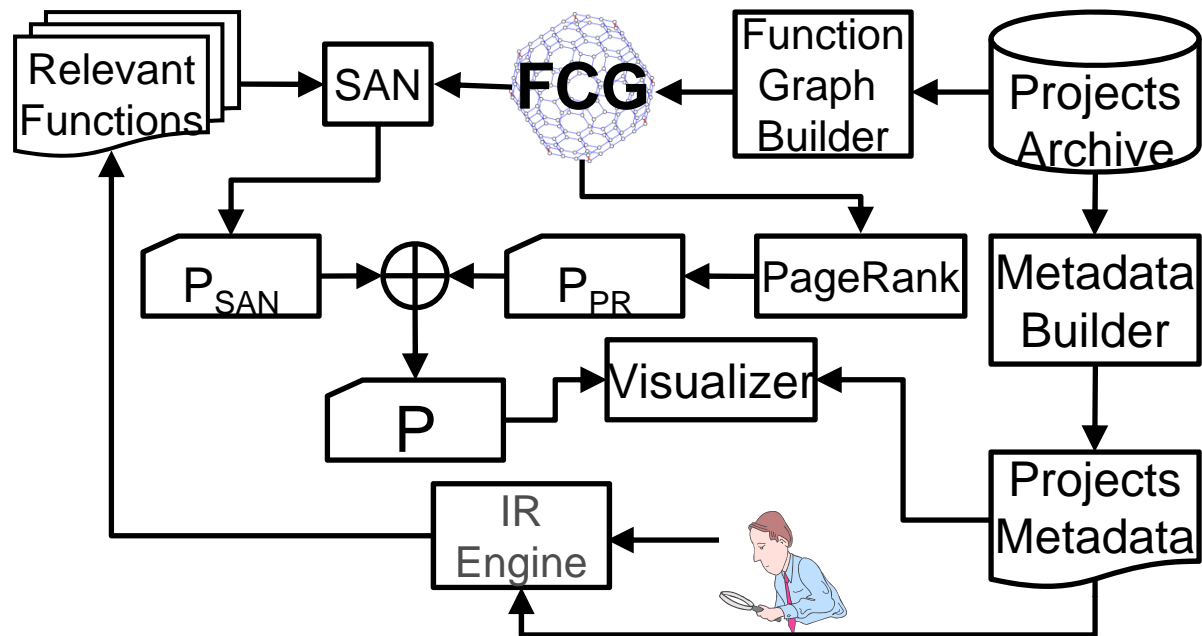


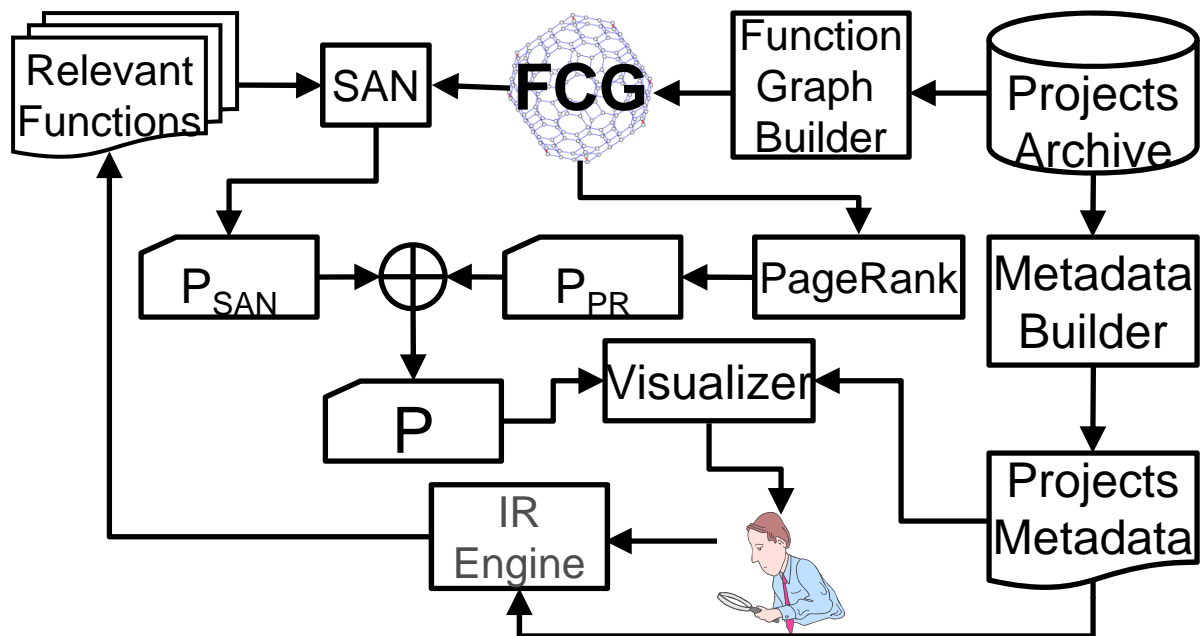


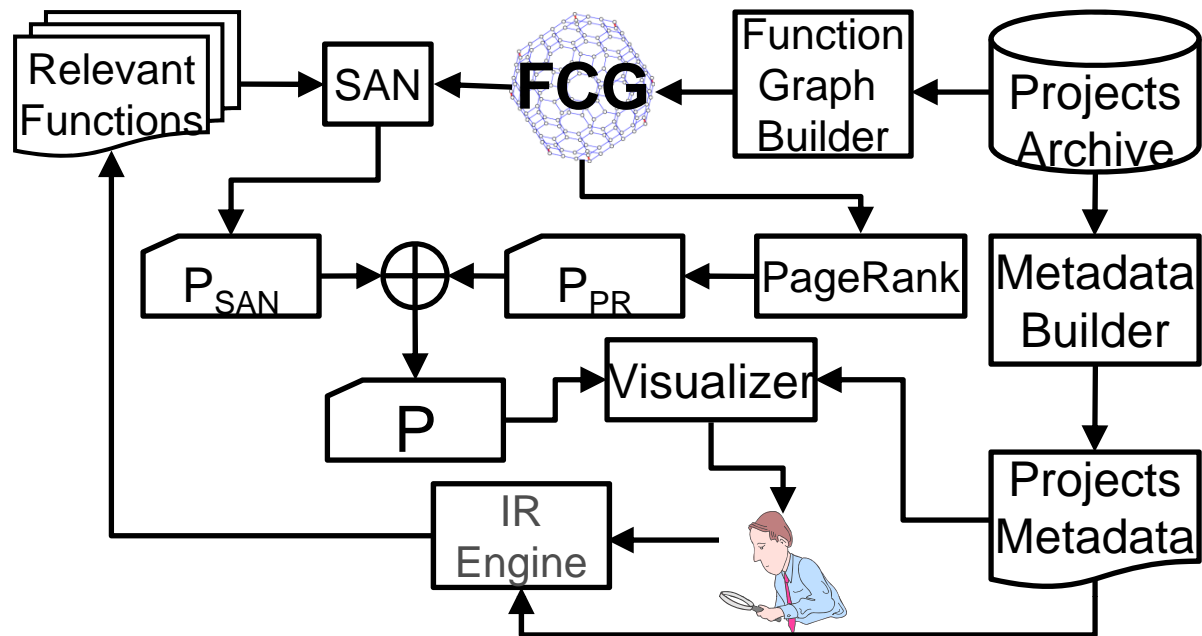






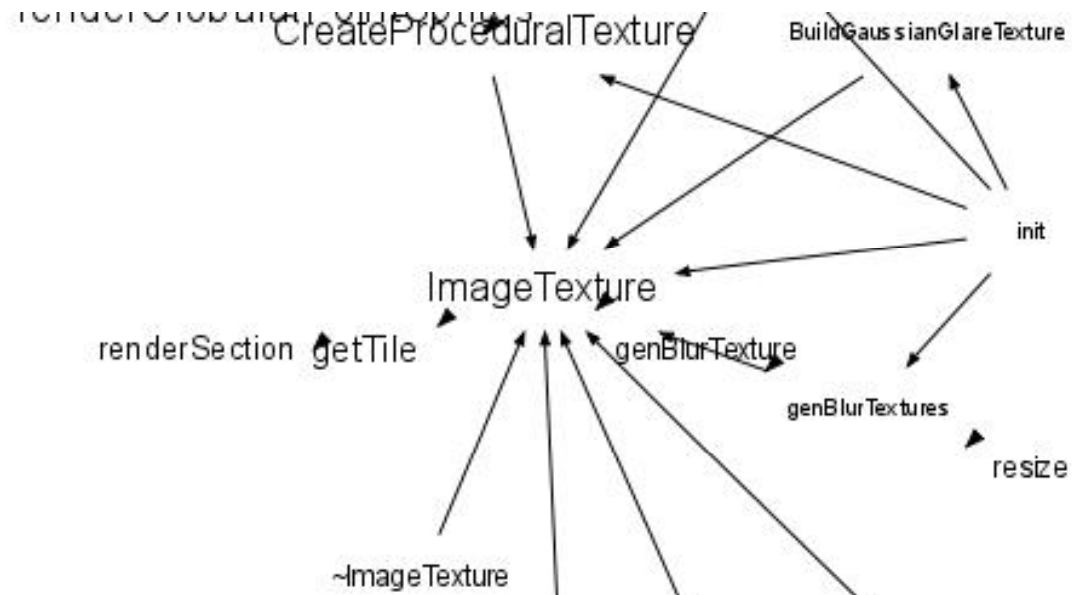






reloadConfiguration  
CreateProceduralTexture  
genBlurTextures  
CreateTextureFromImage  
viewOptionsMenu  
CreateMipMaps  
genBlurTexture  
BuildGaussianGlareTexture  
BuildGaussianDiscTexture  
loadTileTexture  
getTile  
renderGalaxyPointSprites

ultimate  
celestia  
celestia  
celestia  
ultimate  
cs-pseud  
celestia  
celestia  
celestia  
celestia  
celestia  
celestia



# Software Archive

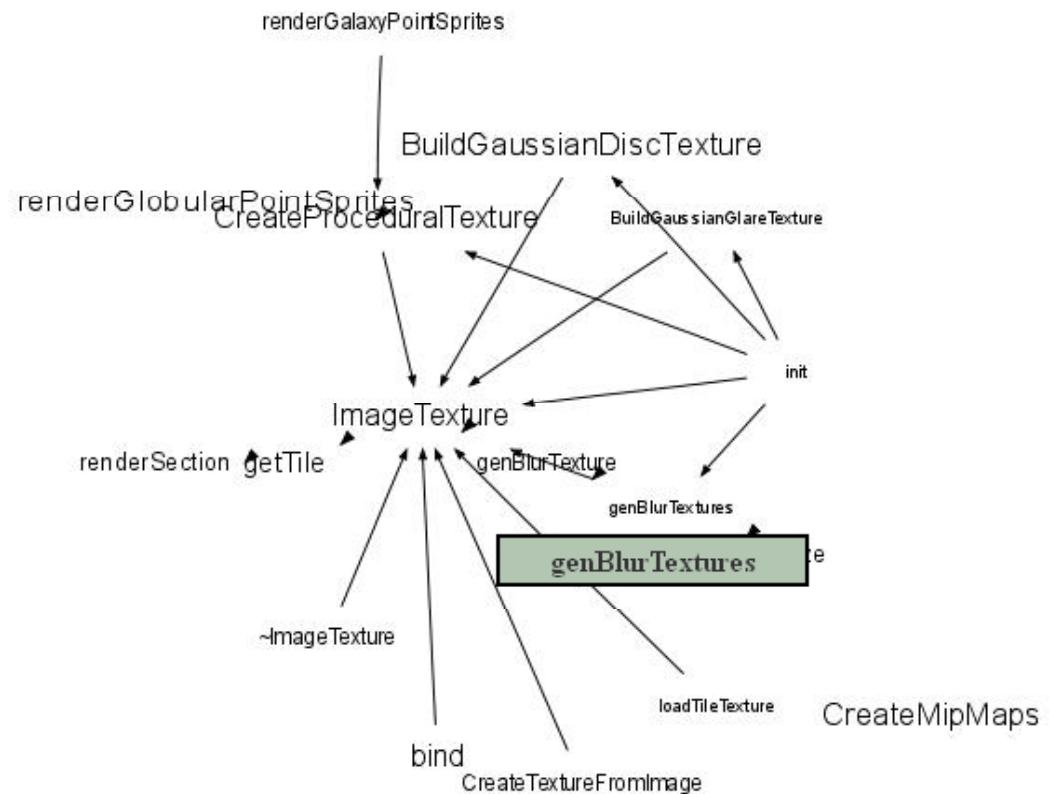
## FreeBSD Ports

- 18,203 C/C++ Projects
- 2.4 Million Files
- 8.5 Million Functions
- 32 Million Function Calls
- 270 Million Total Lines of Code

# Portfolio Interface

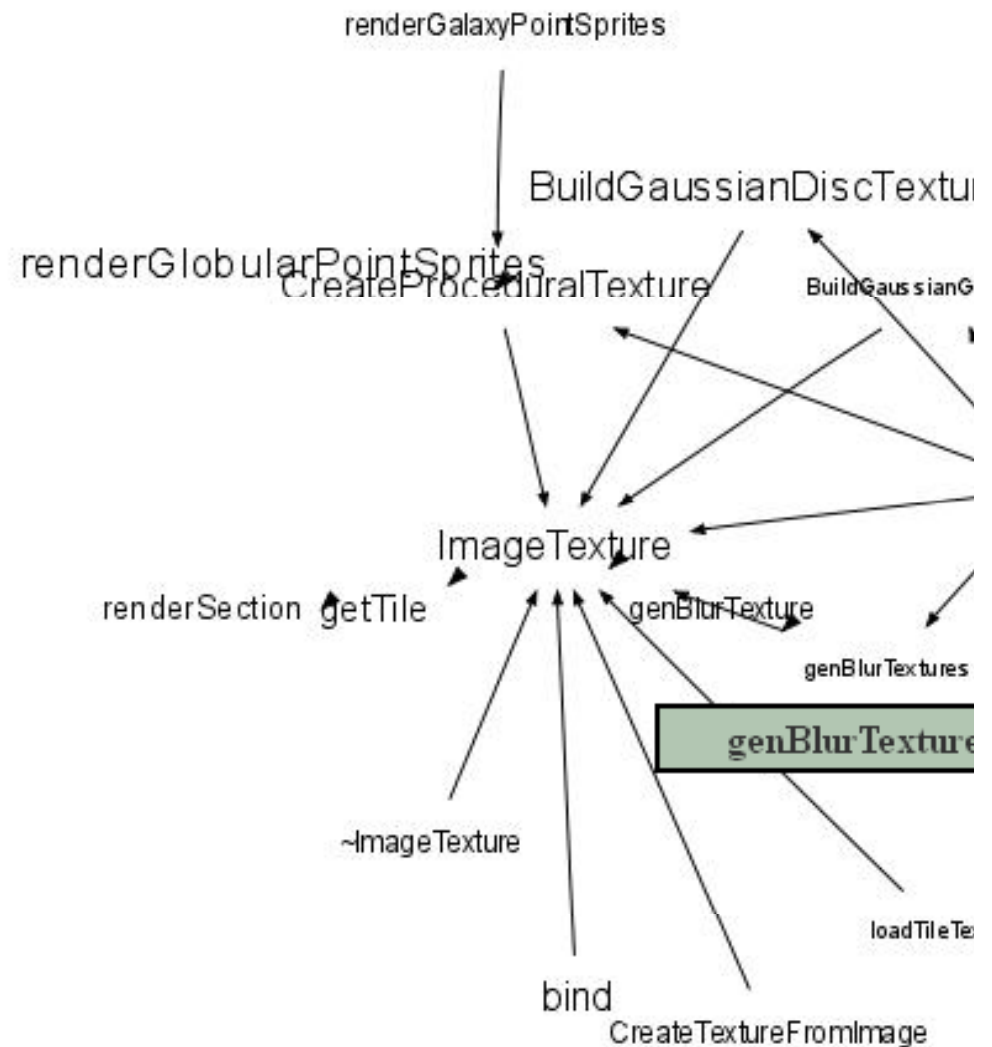
“mip map dithering texture image graphics”

Function Name	Project
<a href="#">CGameWinSystem</a>	<a href="#">ultimate</a>
<a href="#">CGraphicWorld</a>	<a href="#">ultimate</a>
<a href="#">CWinSystem</a>	<a href="#">ultimate</a>
<a href="#">CRenderer</a>	<a href="#">ultimate</a>
<a href="#">ImageTexture</a>	<a href="#">celestia</a>
<a href="#">CSound</a>	<a href="#">ultimate</a>
<a href="#">reloadConfiguration</a>	<a href="#">ultimate</a>
<a href="#">CreateProceduralTexture</a>	<a href="#">celestia</a>
<a href="#">genBlurTextures</a>	<a href="#">celestia</a>
<a href="#">CreateTextureFromImage</a>	<a href="#">celestia</a>
<a href="#">viewOptionsMenu</a>	<a href="#">ultimate</a>
<a href="#">CreateMipMaps</a>	<a href="#">cs-pseud</a>
<a href="#">genBlurTexture</a>	<a href="#">celestia</a>
<a href="#">BuildGaussianGlareTexture</a>	<a href="#">celestia</a>
<a href="#">BuildGaussianDiscTexture</a>	<a href="#">celestia</a>
<a href="#">loadTileTexture</a>	<a href="#">celestia</a>
<a href="#">getTile</a>	<a href="#">celestia</a>
<a href="#">renderGalaxyPointSprites</a>	<a href="#">celestia</a>
<a href="#">renderSection</a>	<a href="#">celestia</a>
<a href="#">renderGlobularPointSprites</a>	<a href="#">celestia</a>
<a href="#">Init</a>	<a href="#">vdrift-2</a>
<a href="#">resize</a>	<a href="#">celestia</a>
<a href="#">init</a>	<a href="#">celestia</a>
<a href="#">~ImageTexture</a>	<a href="#">celestia</a>
<a href="#">bind</a>	<a href="#">celestia</a>



Function `genBlurTextures` is in file `ports_20k/celestia/celestia-1.6.0/src/celengine/render.cpp`.

Function Name	Project
<u>CGameWinSystem</u>	<u>ultimate</u>
<u>CGraphicWorld</u>	<u>ultimate</u>
<u>CWinSystem</u>	<u>ultimate</u>
<u>CRenderer</u>	<u>ultimate</u>
<u>ImageTexture</u>	<u>celestia</u>
<u>CSound</u>	<u>ultimate</u>
<u>reloadConfiguration</u>	<u>ultimate</u>
<u>CreateProceduralTexture</u>	<u>celestia</u>
<u>genBlurTextures</u>	<u>celestia</u>
<u>CreateTextureFromImage</u>	<u>celestia</u>
<u>viewOptionsMenu</u>	<u>ultimate</u>
<u>CreateMipMaps</u>	<u>cs-pseud</u>
<u>genBlurTexture</u>	<u>celestia</u>
<u>BuildGaussianGlareTexture</u>	<u>celestia</u>
<u>BuildGaussianDiscTexture</u>	<u>celestia</u>
<u>loadTileTexture</u>	<u>celestia</u>
<u>getTile</u>	<u>celestia</u>
<u>renderGalaxyPointSprites</u>	<u>celestia</u>
<u>renderSection</u>	<u>celestia</u>
<u>renderGlobularPointSprites</u>	<u>celestia</u>
<u>Init</u>	<u>vdraft-2</u>
<u>resize</u>	<u>celestia</u>
<u>init</u>	<u>celestia</u>
<u>~ImageTexture</u>	<u>celestia</u>
<u>bind</u>	<u>celestia</u>



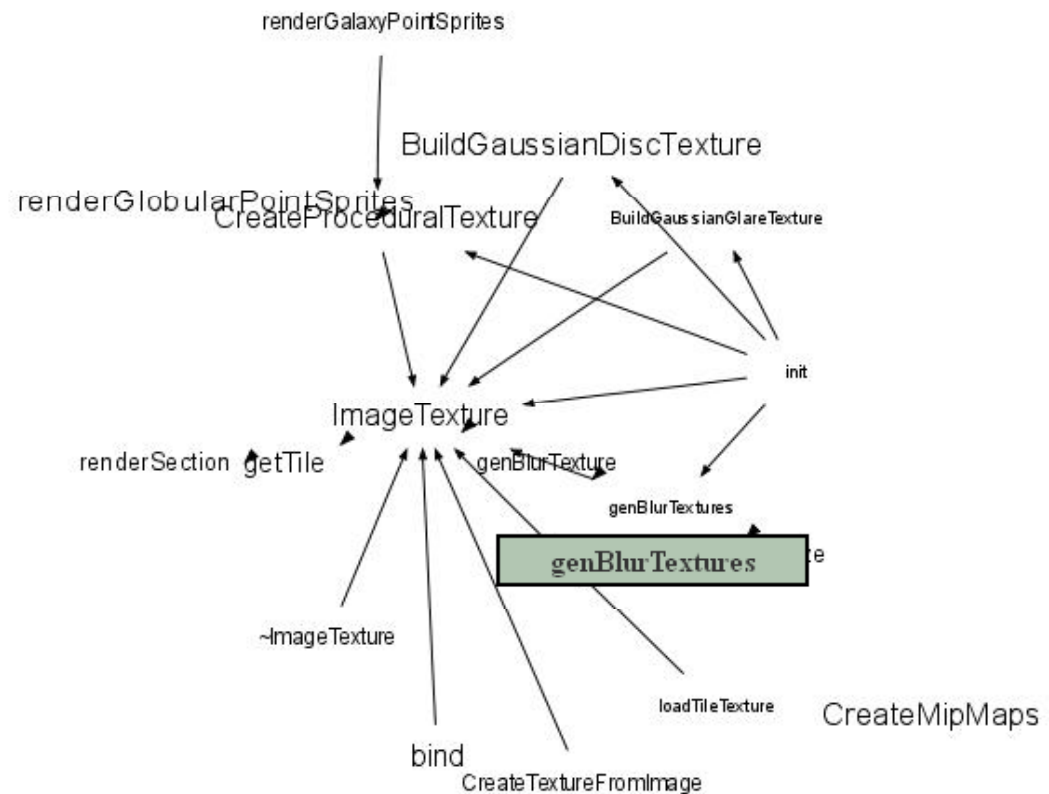
Function **genBlurTextures** is in file **ports\_20k/celestia/celestia-1.6.0**



# Portfolio Interface

“mip map dithering texture image graphics”

Function Name	Project
<a href="#">CGameWinSystem</a>	<a href="#">ultimate</a>
<a href="#">CGraphicWorld</a>	<a href="#">ultimate</a>
<a href="#">CWinSystem</a>	<a href="#">ultimate</a>
<a href="#">CRenderer</a>	<a href="#">ultimate</a>
<a href="#">ImageTexture</a>	<a href="#">celestia</a>
<a href="#">CSound</a>	<a href="#">ultimate</a>
<a href="#">reloadConfiguration</a>	<a href="#">ultimate</a>
<a href="#">CreateProceduralTexture</a>	<a href="#">celestia</a>
<a href="#">genBlurTextures</a>	<a href="#">celestia</a>
<a href="#">CreateTextureFromImage</a>	<a href="#">celestia</a>
<a href="#">viewOptionsMenu</a>	<a href="#">ultimate</a>
<a href="#">CreateMipMaps</a>	<a href="#">cs-pseud</a>
<a href="#">genBlurTexture</a>	<a href="#">celestia</a>
<a href="#">BuildGaussianGlareTexture</a>	<a href="#">celestia</a>
<a href="#">BuildGaussianDiscTexture</a>	<a href="#">celestia</a>
<a href="#">loadTileTexture</a>	<a href="#">celestia</a>
<a href="#">getTile</a>	<a href="#">celestia</a>
<a href="#">renderGalaxyPointSprites</a>	<a href="#">celestia</a>
<a href="#">renderSection</a>	<a href="#">celestia</a>
<a href="#">renderGlobularPointSprites</a>	<a href="#">celestia</a>
<a href="#">Init</a>	<a href="#">vdrift-2</a>
<a href="#">resize</a>	<a href="#">celestia</a>
<a href="#">init</a>	<a href="#">celestia</a>
<a href="#">~ImageTexture</a>	<a href="#">celestia</a>
<a href="#">bind</a>	<a href="#">celestia</a>

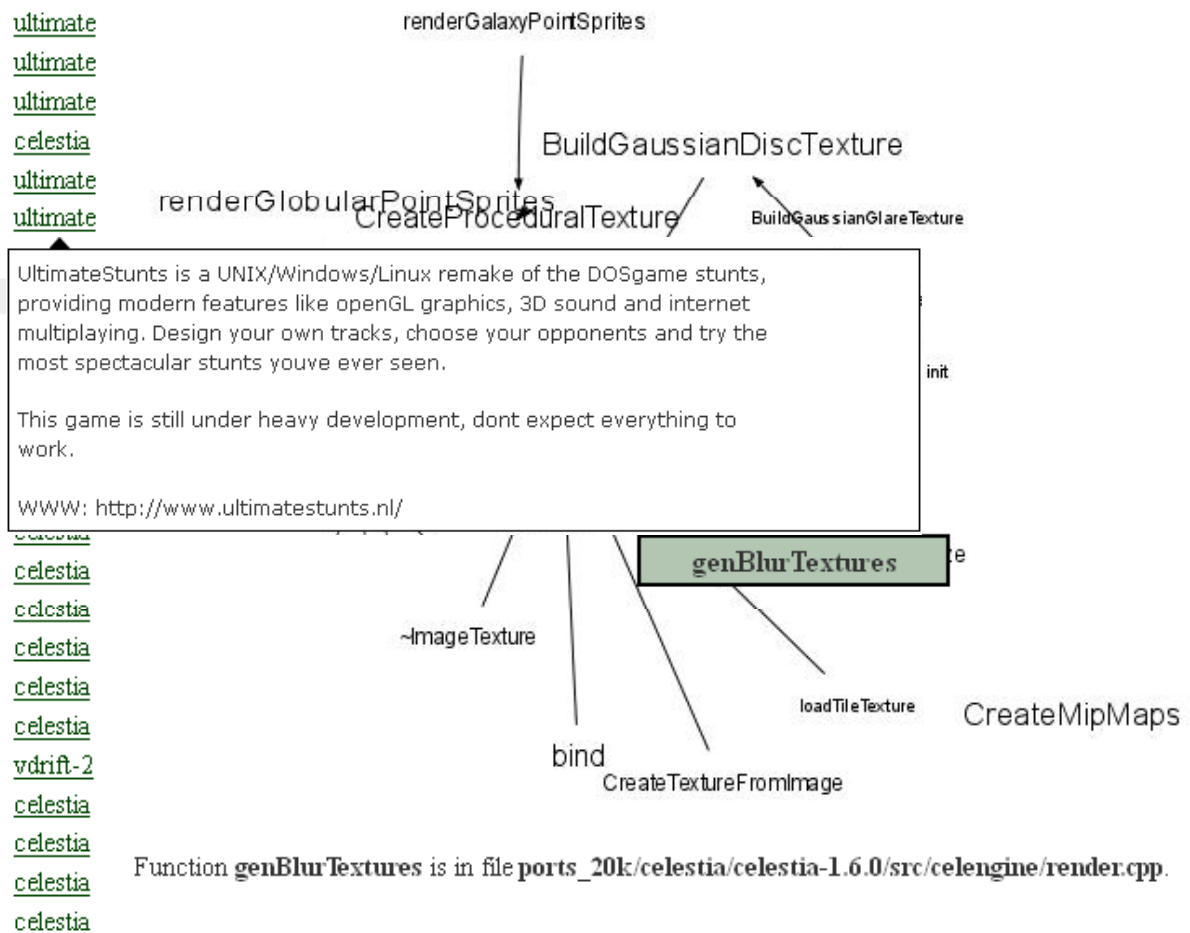


Function `genBlurTextures` is in file `ports_20k/celestia/celestia-1.6.0/src/celengine/render.cpp`.

# Portfolio Interface

“mip map dithering texture image graphics”

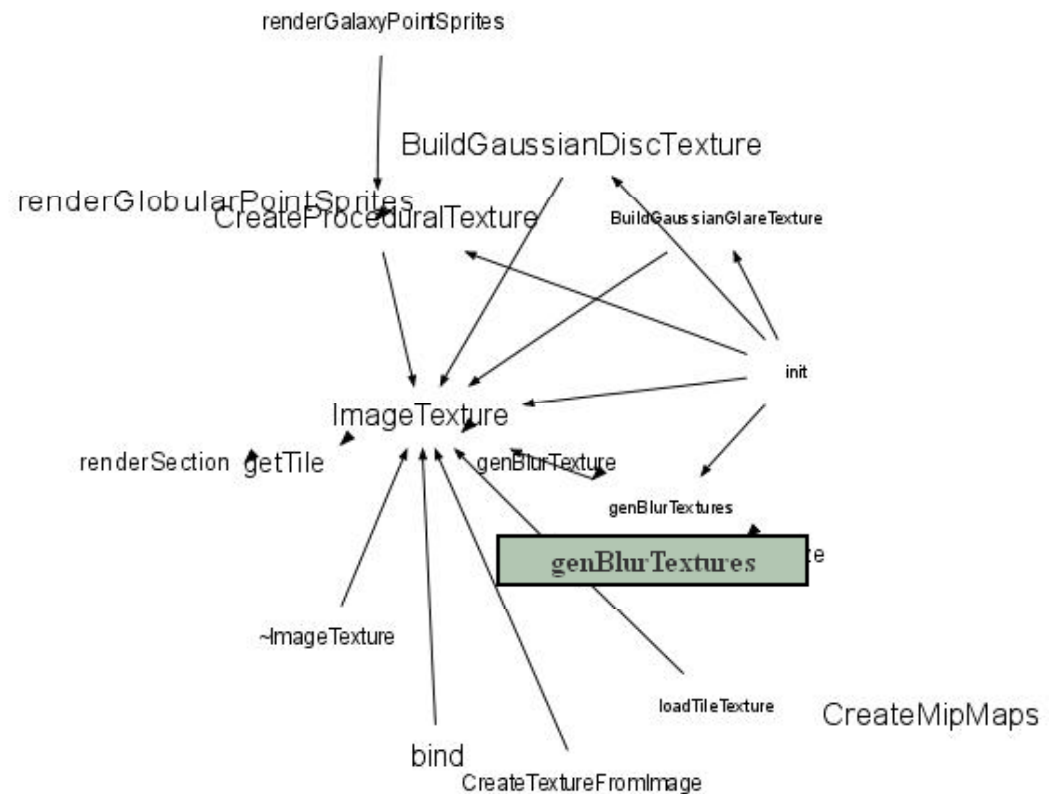
Function Name	Project
<a href="#">CGameWinSystem</a>	<a href="#">ultimate</a>
<a href="#">CGraphicWorld</a>	<a href="#">ultimate</a>
<a href="#">CWinSystem</a>	<a href="#">ultimate</a>
<a href="#">CRenderer</a>	<a href="#">ultimate</a>
<a href="#">ImageTexture</a>	<a href="#">celestia</a>
<a href="#">CSound</a>	<a href="#">ultimate</a>
<a href="#">reloadConfiguration</a>	<a href="#">ultimate</a>
<a href="#">CreateProceduralTexture</a>	
<a href="#">genBlurTextures</a>	
<a href="#">CreateTextureFromImage</a>	
<a href="#">viewOptionsMenu</a>	
<a href="#">CreateMipMaps</a>	
<a href="#">genBlurTexture</a>	
<a href="#">BuildGaussianGlareTexture</a>	
<a href="#">BuildGaussianDiscTexture</a>	
<a href="#">loadTileTexture</a>	<a href="#">celestia</a>
<a href="#">getTile</a>	<a href="#">celestia</a>
<a href="#">renderGalaxyPointSprites</a>	<a href="#">celestia</a>
<a href="#">renderSection</a>	<a href="#">celestia</a>
<a href="#">renderGlobularPointSprites</a>	<a href="#">celestia</a>
<a href="#">Init</a>	<a href="#">vdrift-2</a>
<a href="#">resize</a>	<a href="#">celestia</a>
<a href="#">init</a>	<a href="#">celestia</a>
<a href="#">~ImageTexture</a>	<a href="#">celestia</a>
<a href="#">bind</a>	<a href="#">celestia</a>



# Portfolio Interface

“mip map dithering texture image graphics”

Function Name	Project
<a href="#">CGameWinSystem</a>	<a href="#">ultimate</a>
<a href="#">CGraphicWorld</a>	<a href="#">ultimate</a>
<a href="#">CWinSystem</a>	<a href="#">ultimate</a>
<a href="#">CRenderer</a>	<a href="#">ultimate</a>
<a href="#">ImageTexture</a>	<a href="#">celestia</a>
<a href="#">CSound</a>	<a href="#">ultimate</a>
<a href="#">reloadConfiguration</a>	<a href="#">ultimate</a>
<a href="#">CreateProceduralTexture</a>	<a href="#">celestia</a>
<a href="#">genBlurTextures</a>	<a href="#">celestia</a>
<a href="#">CreateTextureFromImage</a>	<a href="#">celestia</a>
<a href="#">viewOptionsMenu</a>	<a href="#">ultimate</a>
<a href="#">CreateMipMaps</a>	<a href="#">cs-pseud</a>
<a href="#">genBlurTexture</a>	<a href="#">celestia</a>
<a href="#">BuildGaussianGlareTexture</a>	<a href="#">celestia</a>
<a href="#">BuildGaussianDiscTexture</a>	<a href="#">celestia</a>
<a href="#">loadTileTexture</a>	<a href="#">celestia</a>
<a href="#">getTile</a>	<a href="#">celestia</a>
<a href="#">renderGalaxyPointSprites</a>	<a href="#">celestia</a>
<a href="#">renderSection</a>	<a href="#">celestia</a>
<a href="#">renderGlobularPointSprites</a>	<a href="#">celestia</a>
<a href="#">Init</a>	<a href="#">vdrift-2</a>
<a href="#">resize</a>	<a href="#">celestia</a>
<a href="#">init</a>	<a href="#">celestia</a>
<a href="#">~ImageTexture</a>	<a href="#">celestia</a>
<a href="#">bind</a>	<a href="#">celestia</a>



Function `genBlurTextures` is in file `ports_20k/celestia/celestia-1.6.0/src/celengine/render.cpp`.

	<u>ultimate</u>
	<u>ultimate</u>
	<u>ultimate</u>
	<u>ultimate</u>
	<u>celestia</u>
	<u>ultimate</u>
	<u>ultimate</u>
	<u>celestia</u>
	<u>celestia</u>

celestia

celestia

celestia

celestia

celestia

celestia

celestia

celestia

celestia

celestia

celestia

celestia

celestia

celestia

celestia

celestia

celestia

celestia

celestia

celestia

celestia

celestia

celestia

celestia



Function `genBlurTextures` is in file `ports_20k/celestia/celestia-1.6.0/src/celengine/render.cpp`.

# Experiment

To compare **Portfolio** with **Google Code Search** and **Koders**

49 C/C++ Programmers participated

- 44 Professionals from Accenture
- 5 Students from the University of Illinois at Chicago

# Large Case Studies are Rare

“First, it is very difficult to scale human experiments to get quantitative, significant measures of usefulness; this type of large-scale human study is very rare.

Second, comparing different recommenders using human evaluators would involve carefully designed, time-consuming experiments; this is also extremely rare.”

Saul, Filkov, Devanbu, Bird  
Recommending Random Walks, ESEC/FSE'07



## Participants' Role

- 1) Receive Task and Search Engine

**Write a utility for dithering mip map images that are used for rendering texture.**

- 2) Translate Task to Query, enter into Engine

### Portfolio

A Source Code Search Engine for [FreeBSD ports](#)

 Like

7

# Likert Scale - Confidence

- ★☆☆☆☆ Completely irrelevant – there is absolutely nothing that the participant can use from this retrieved code fragments, nothing in it is related to keywords that the participant chose based on the descriptions of the tasks.
- ★★☆☆☆ Mostly irrelevant – a retrieved code fragment is only remotely relevant to a given task; it is unclear how to reuse it.
- ★★★☆☆ Mostly relevant – a retrieved code fragment is relevant to a given task and participant can understand with some modest effort how to reuse it to solve a given task.
- ★★★★★ Highly relevant – The participant is highly confident that code fragment can be reused and s/he clearly see how to use it.



# Analysis of the Results

Metrics:

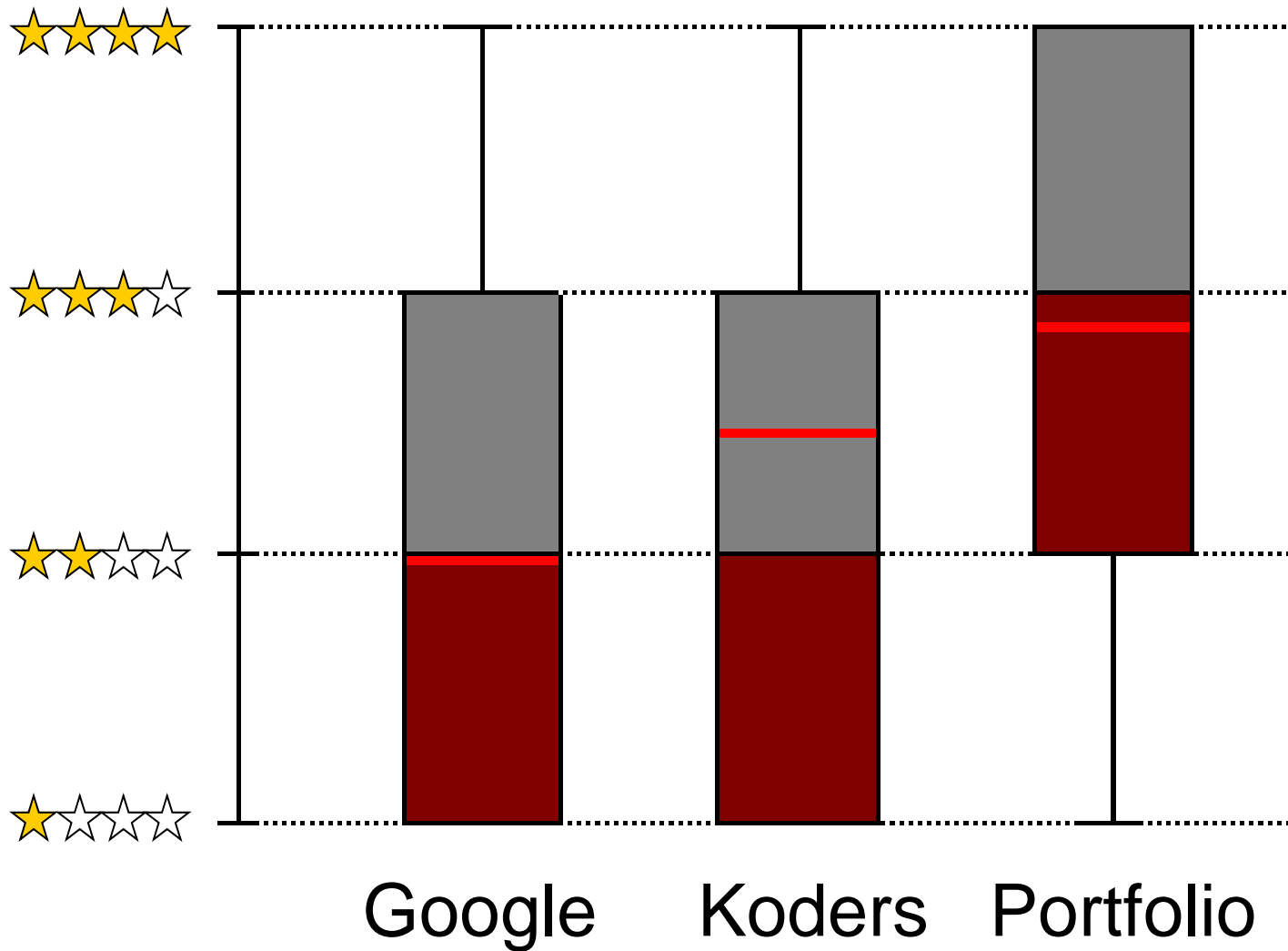
Confidence (C)

Precision (P)

Normalized Discounted Cumulative Gain (NG)

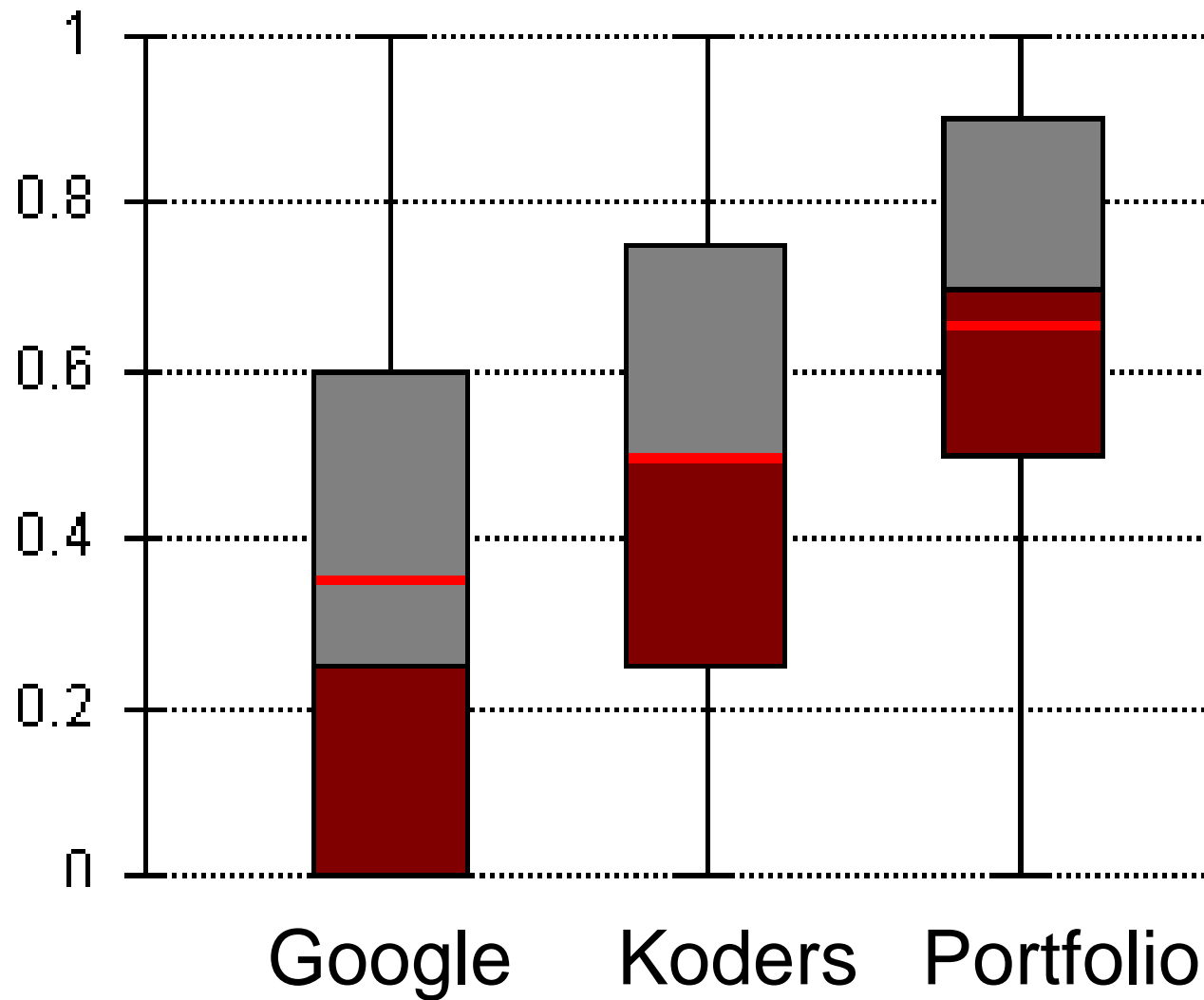
Search Engine	Queries Entered	Responses Rated
Portfolio	184	1276
Google Code Search	198	1373
Koders	208	1486

# Results – Confidence



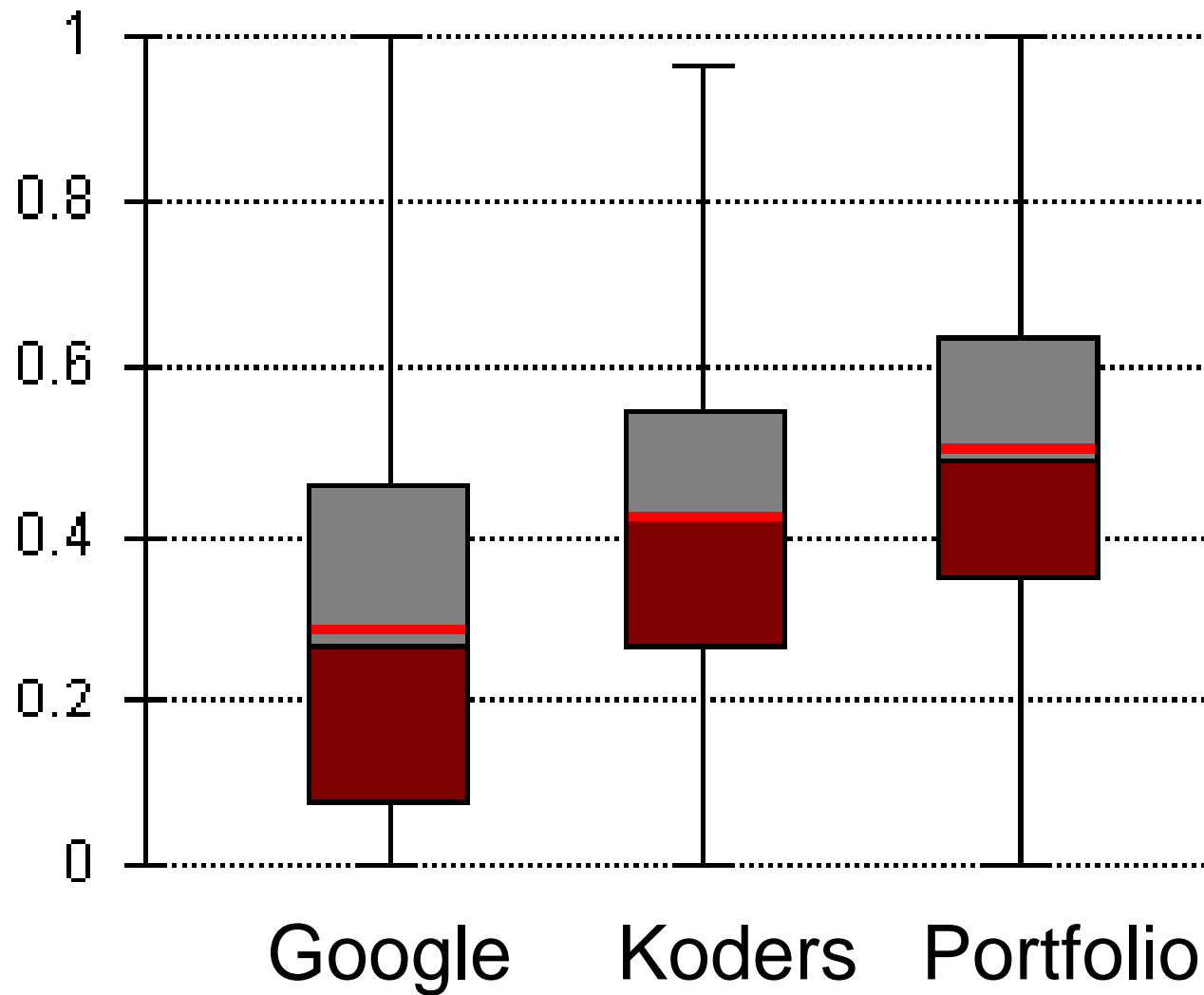
p	$<5.0 \cdot 10^{-108}$
F	261.3
F <sub>crit</sub>	3.01

## Results – Precision



p	$<8.6 \cdot 10^{-22}$
F	52.5
F <sub>crit</sub>	3.01

# Results – Normalized Discounted Gain



p	$<2.5 \cdot 10^{-18}$
F	43.8
F <sub>crit</sub>	3.01

# Statistical Analysis – ANOVA

Null Hypothesis rejected in all cases:

$H_0$  – There is no difference in the C, P, or NG mean values among users of Portfolio, Google Code Search, and Koders.

$H_1$  – There is statistically-significant difference in the numbers of C, P, and NG mean values among users of Portfolio, Google Code Search, and Koders.

Metric	p	F	F <sub>critical</sub>
Confidence	$< 5.0 \cdot 10^{-108}$	261.3	3.01
Precision	$< 8.6 \cdot 10^{-22}$	52.5	3.01
Discounted Gain	$< 2.5 \cdot 10^{-18}$	43.8	3.01

# Statistical Analysis – ANOVA

Null Hypothesis rejected in all cases:

~~$H_0$  – There is no difference in the C, P, or NG mean values among users of Portfolio, Google Code Search, and Koders.~~

$H_1$  – There is statistically-significant difference in the numbers of C, P, and NG mean values among users of Portfolio, Google Code Search, and Koders.

Metric	p	F	F <sub>critical</sub>
Confidence	$< 5.0 \cdot 10^{-108}$	261.3	3.01
Precision	$< 8.6 \cdot 10^{-22}$	52.5	3.01
Discounted Gain	$< 2.5 \cdot 10^{-18}$	43.8	3.01

# Programmer Experience Relations

Do more-experienced programmers report different results than less-experienced programmers?

Null Hypothesis **not** rejected:

$H_0$  – There is no difference in the C, P, or NG mean values among experienced and less-experienced users of Portfolio, Google Koder Search, and Koders.

$H_1$  – There is statistically-significant difference in the numbers of C, P, and NG mean values among experienced and less-experienced users of Portfolio, Google Code Search, and Koders.

# Programmer Experience Relations

Do more-experienced programmers report different results than less-experienced programmers?

Null Hypothesis **not** rejected:

$H_0$  – There is no difference in the C, P, or NG mean values among experienced and less-experienced users of Portfolio, Google Koder Search, and Koders.

~~$H_1$  – There is statistically-significant difference in the numbers of C, P, and NG mean values among experienced and less-experienced users of Portfolio, Google Code Search, and Koders.~~



## Responses from Programmers

“The search engine Portfolio is a good search tool ... developers won't waste time exploring different projects or functions.”

“Portfolio looks into functional not based exactly on the wording like Google, so when it's found the right function, the search is really on target.”

“The ‘code web’ of search results was very helpful for finding out which things to analyze.”

# Suggestions from Programmers

“The best addition to Portfolio would be the ability to navigate through functions much like an IDE can.”

“I would like to see more feedback from the search and more options on how to search.”

“If query is misspelled, [Portfolio] does not return suggestions.”

# Ongoing Improvements

## Data Available

- All Source Code in Our Repository

- Function Dependency Extractor: FUNDEX

- Case Study Tasks and Responses

## Programmer Access

- SOAP web service

- Java search now available


See <http://www.searchportfolio.net/>

# Visit us online and at Facebook!

## <http://www.searchportfolio.net/>





**Portfolio**  
Wall Info Photos Discussions +

Share:  Status  Question  Photo  Link  Video

Write something...

Portfolio + Others Just Portfolio Just Others Spam Settings

**Portfolio** Portfolio is now accessible via SOAP as a web service! As the API grows, we will post howtos and examples to this page:  
[Portfolio Programmer Access Guide](#)  
[www.cs.wm.edu](http://www.cs.wm.edu)  
October 12, 2010 at 9:45am · Like · Comment · Share · Promote

**Portfolio** Download our tool to create and explore your own call graph of thousands of projects!  
[http://www.cs.wm.edu/semeru/portfolio/fundex\\_v1.0.tar.gz](http://www.cs.wm.edu/semeru/portfolio/fundex_v1.0.tar.gz)  
[www.cs.wm.edu](http://www.cs.wm.edu)  
October 11, 2010 at 12:02pm · Like · Comment · Share · Promote

**Portfolio** We conducted an experiment with 49 C/C++ programmers, including 44 professionals from Accenture. We asked all participants to compare 3 source code search engines: Google Code Search, Koders, and Portfolio. We found statistically-significant improvement of Portfolio over both GCS and Koders! From this link you can down...  
[See More](#)  
<http://www.cs.wm.edu/semeru/portfolio/ExperimentMaterials.tar.gz>  
[www.cs.wm.edu](http://www.cs.wm.edu)  
October 11, 2010 at 12:00pm · Like · Comment · Share · Promote

**Portfolio** The results of the query "record music file". On the left are individual relevant functions (and projects) from FreeBSD Ports. On the right is a call graph showing how those functions are connected!  
  
[Using Portfolio](#)

Edit Page  
Promote with an Ad  
Suggest to Friends  
Remove from My Page's Favorites

Write something about Portfolio.

**Insights**  
[See All](#)

25 Monthly Active Users  
0 Daily New Likes  
4 Daily Post Views  
0 Daily Post Feedback

Insights are visible to page admins only.

**38 Friends Like This**  
6 of 38 Friends [See All](#)

  
Nimit Shah Isabelle Perseil Stefano Crespi

  
Christoph Csallner Harvey Siy S.C. Cheung

**52 People Like This**  
[See All](#)



**Portfolio**  
A Source Code Search Engine for [FreeBSD ports](#)  
 Like 8

### Free source code for everyone!

Search for code among 18,203 C/C++ projects.

Welcome to Portfolio. From this page you can search for functions in C or C++ that perform a task you specify. You will see relevant functions along with a [call graph](#) showing how they interact. All of the code indexed by Portfolio comes from [FreeBSD's](#) source code repository [ports](#), and includes over 18,000 C/C++ projects, 2.2 million files, and 8.5 million functions!

Query:   
How many results to display:

### Programmer access to Portfolio:

You can now build Portfolio into your own programs using our SOAP web service. Visit the [programmer access guide](#) for more information. **NEW!**

### How Portfolio works:

There are essentially two parts to the system. One downloads, extracts, and parses the source code from ports. The parsing involves locating every project in the archive, then every C/C++ file in every project, and then every function in every file. Each function is then analyzed to see all the other functions that it calls.

The second part is the search engine you can use here. First, keywords from your query are matched to keywords in the source code of functions (such as identifier names, comments, or the contents of strings). Second, we find functions which call the functions from the first step. Third, we consider the global importance of the functions of both steps by looking at the [PageRank](#) score of each function. When calculating the PageRank, we look at the call graph of *every* function and function call.