Overview

- Quick descriptio9 of VR Juggler
- Major features of VR Juggler
- Descriptio9 of major contributio9s
- Conclusio9s

What is VR Juggler?

- Open Source VR software development environment
- VR virtual platform & application framework
- Developed at Iowa State University
  - Project director: Dr. Carolina Cruz-

6 Dn.C 0.75 -34.5 5 0.75170 0 rgf 0.c 0 Tw (–) Tj -0.75 00.75 0.75 TD 0.2 0.8 0.2 rg (–) Tj 18.75 -0

VR Juggler features

Major contributions

- Virtual platform
- Modular extensible micro-kernel system
- Application   objects

VP29 TMgler Kernel

Kernel subsystem

Microkernel architecture

- Controls the entire system
- Manages communication
- Internal Managers
  - „77 0 0 0 r3280.036 Tontfuncanagality beyond 1 rgscope

Applications objects

Benefits

inte

- Single

What is run

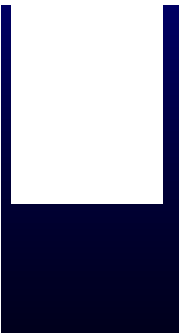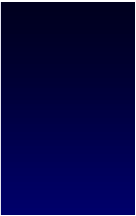# Reconfiguration benefits

System setup and configuration

- Easier initial setup using interactive testing
  -

Conclusions and future workConclus

Conclusions

-

Questions??
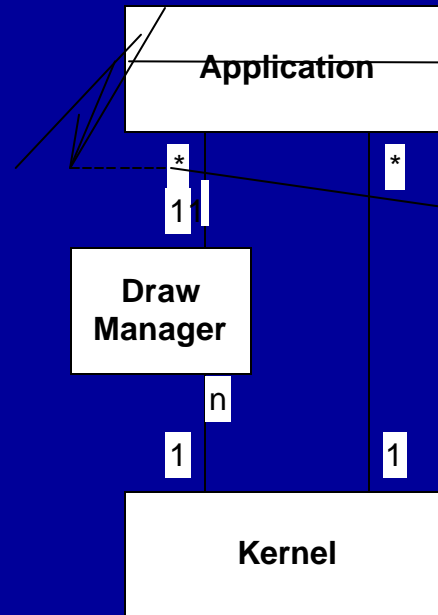
Other

Input Manager

Display manager

- Contains all information about display windows
- Performs all vi422.2 calcultion

Misc. internal managers

Display Manager

- Contains all d81f-nMnhation(•) Tj -0.75 0.75151515
    - re)8Twfiguration requests(•) Tj -0.75 0.751515151
        - • •

Draw manager

Application

Draw Manager

Kernel

- Provides external interface for applications
- Allows access to API specific functionality
- Handles all details of the API

Other ExternaE managers

Parameterization of components

Chunk: Window

Size	Int	Int

- Juggler "config chunks" specify single unit of configuration
- Parameterize all configurable parts of system
- Edited with Java base GUI (vjControl)

Application interface

vjApp

+init()
+apiInit()
+preFrame()
+intraFrame()
+postFrame()
+...()

vjGlApp

+draw()
+contextInit()
+contextPreDraw()
+...()

myApp

+init()
+apiInit()
+preFrame()
+intraFrame()
+postFrame()
+draw()
+contextInit()
+contextPreDraw()
+...()

- Other…
  - Reconfiguration
  - Reset
  - Exit
  - focus change

Call timing

```
int main(int argc, char* argv[])
{
    vjKernel* kernel = vjKernel::instance();     // Get the kernel
    simpleApp* app = new simpleApp();            // Create app object

    simkernel->loadConfigFile(...);                   // Configure  kernel

    simkernel->start();                          // Start the kernel thread

    kernel->setApplication(app);      // Give applicaton to kernel
```

- setApp starts the kernel calling the application's member functions