Collision Detection and Teamcenter Haptics: CATCH

Project Plan

<u>Group: May 14-30</u>

Anthony Alleven	aalleven@iastate.edu
James Erickson	jamese@iastate.edu
Paul Uhing	pfuhing@iastate.edu
Logan Scott	logscott@iastate.edu
Matt Mayer	mmayer@iastate.edu

1. Problem/Need

The field of virtual reality is advancing in leaps and bounds. One blocking factor to virtual reality's progression is a lack of reality. Users need to have some feedback on their actions. Haptic feedback is one way to provide users with feedback. An application that provides users with haptic feedback for their actions in the virtual world is needed.

While Dr. Vance's group has already demonstrated the ability to solve the above problem, their solution is not commercially viable. Our project is to make this project more commercially viable by utilizing commercial software.

2. Solution

Create a prototype standalone service to demonstrate manipulation of 3D models using haptics and Teamcenter Visualization. This program will perform collision detection and 3D transformation on object models. This project is important because this system has been previously created using in-house software not suited to commercial use. This project seeks to create a commercial-compatible solution to the problem stated above.

3. Concept Sketch

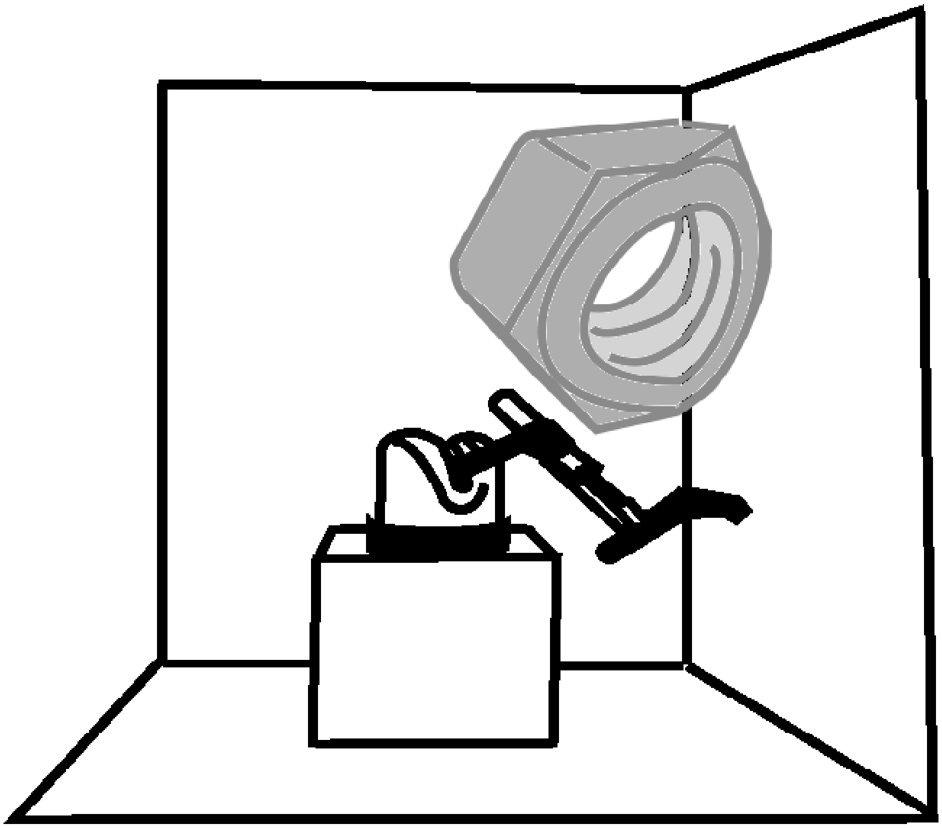
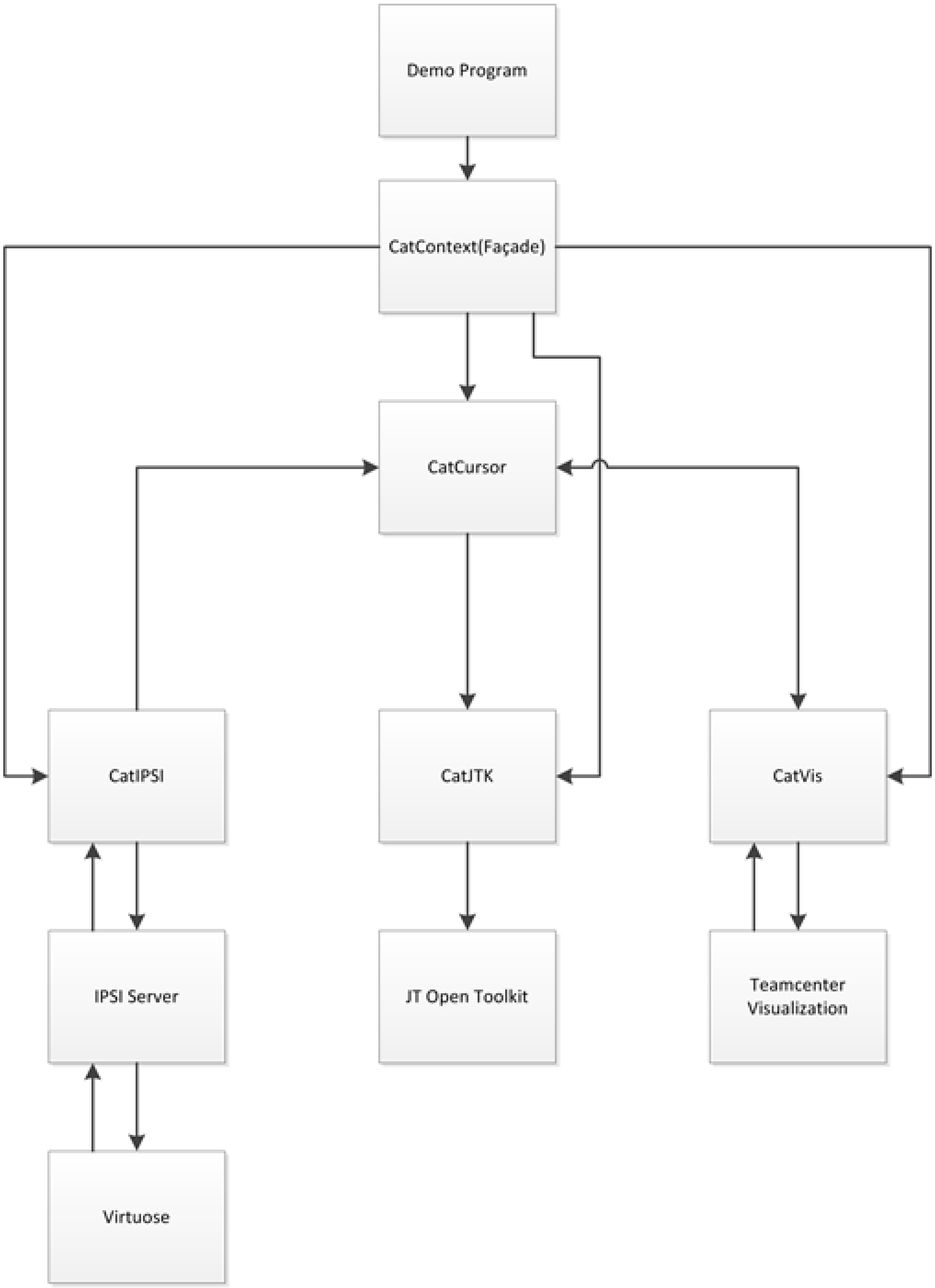


Image: *Concept sketch showing the Virtuose device. An input device with 6-axis motion for manipulating objects in 3D space with the ability to provide haptic feedback. The device is stationed on a static base, in front of a 3D visualization of an example part model.*

4. Block Diagram



5. System Description

Create a standalone service to interface between a haptic device and Teamcenter Visualization (TCVis). This program will perform collision detection and 3D transformations on object models. The resulting feedback from these calculations will be output to the haptic device. Input from the haptic device will trigger these transformations and interactions.

6. Operating environment

The application will run natively as a C++ executable through a Windows command line.

7. User interface description

1. Command-line interface
2. Virtuose Haptic Arm
3. Teamcenter Visualization

8. Functional requirements

1. Manipulate a cursor in Teamcenter with a haptic device
2. Select an object in Teamcenter Visualization with the haptic device
3. Have the object follow rules of physics with appropriate haptic feedback
4. Support loading part geometry via Jupiter Tessellation (JT) files

9. Non-Functional requirements

1. The lag time between input and output shall be less than 200ms
2. All public modules and functions shall be documented to the extent at which they could be recreated by a third party.
3. After accounting for lag time, all object models shall be synchronized.

10. Market and literature survey

Our primary market is all customers of Teamcenter Visualization.
Work on projects similar to this one has been done before. Below are a few papers regarding these similar projects.

A. *A Hybrid method for haptic feedback to support manual virtual product assembly,*
Author: Daniela Faas, 2010 Iowa State
Link: http://lib.dr.iastate.edu/cgi/viewcontent.cgi?article=2468&context=etd

B. *A new type haptics-based virtual environment system for assembly training of complex products.*
Authors: PingJun Xia, António M. Lopes, Maria Teresa Restivo, YingXue Yao
Link: http://link.springer.com/article/10.1007/s00170-011-3381-8#page-1

C. *Desktop haptic virtual assembly physically based modelling,*
Author: Brad M. Howard, Judy M. Vance
Link: http://link.springer.com/article/10.1007/s10055-007-0069-3#page-1

D. *Stable haptic interaction with Virtual Environments,*

Authors: Adams, R.J.

Dept. of Electr. Eng., Washington Univ., Seattle, WA, USA and Hannaford, B.

Link:

http://ieeexplore.ieee.org/xpl/login.jsp?tp=&arnumber=768179&url=http%3A%2F%2Fieeexplore.ieee.org%2Fstamp%2Fstamp.jsp

11. Deliverables

A standalone program should perform at least the following tasks:
a) Load a JT file. b) Define the target part in the JT scene graph. c) Apply transformation(s) to the target part. d) Perform collision detection. e) Send force feedback calculated by the collision detection to the haptic device. f) Send the updated transformation matrix (based on collision detection) to Teamcenter Visualization to update the scene.

12. Work Plan:

The Agile software development methodology will be our primary development structure.

Work breakdown structure

Module	Description	Responsibility
CatContext	Façade module to interact with outside programs	Paul Uhing
CatJtk	Parses JT files to extract triangle geometry for Physics Engine. Uses JT Toolkit library	Logan Scott
CatIPSI	Interacts with existing physics engine (IPSI).	James Erickson
CatVis	Interacts with Teamcenter Visualizer through VisController library	Anthony Alleven
CatCursor	Main control loop	Matt Mayer

Resource Requirements

1. Access to all required APIs
2. Access to VRAC
3. Access to the haptic arm

Risks

Risk	**Mitigation**
CPU requirements for collision detection and I/O blocking	Support remote connection to IPSI physics engine, allowing IPSI processing to be done on separate computer
Data Bandwidth	Throttle-able refresh rate
IPSI Model Representations vs. Teamcenter Model Representations	Decoupled modules with abstract data model when transferring geometry
Two-way communication may not be available before project completion	Either (a) support forced selection of parts or (b) implement part selection via IPSI

Project Timeline

Our live timeline can be reached at:
http://publish.smartsheet.com/c3063fe887cf422e83c6c1a207c264dd

The table below represents a copy of the timeline at the time of writing. However, the table accessible via the URL above is well-formatted and will be updated over time.

Task Name	Duration	Start	Finish	Predecessors	Assigned To
Spark	7	10/13/13	10/19/13		
Research and Demonstrate receiving Data from Virtuose	7	10/13/13	10/19/13		Paul Uhing
Document Matrix Representations between APIs and Main program	7	10/13/13	10/19/13		James C Erickson
Research and Demonstrate opening a model file using JTopenTk	7	10/13/13	10/19/13		Logan Scott
Research and Demonstrate Converting position data from IPSI to transformation Matrix	7	10/13/13	10/19/13		Matt Mayer
Layout Overall Activity Diagram	7	10/20/13	10/26/13	1	Paul Uhing
Define Module interfaces	7	10/27/13	11/02/13	6	
Define TCVisController Module Interaction IN/OUT	7	10/27/13	11/02/13	6	Anthony Alleven
Define Cursor Tracking Module Interface IN/OUT	7	10/27/13	11/02/13	6	Matt Mayer
Define JTopenToolKit Interaction Module IN/OUT	7	10/27/13	11/02/13	6	Logan Scott
Define IPSI Module Interface IN/OUT	7	10/27/13	11/02/13	6	James C Erickson
Define API for Facade Module IN/OUT	7	10/27/13	11/02/13	6	Paul Uhing

Layout Application Skeleton	7	11/03/13	11/09/13	7	Anthony Alleven
Implement facade class	14	11/10/13	11/23/13	13	Paul Uhing
Implement Integration JTopenTk and IPSI modules to describe objects in physics engine	14	11/10/13	11/23/13	13	Logan Scott
Implement Integration between IPSI, VirtuoseAPI, JTopenTK to perform collision detection	14	11/10/13	11/23/13	13	James C Erickson
Implement Modules	21	11/03/13	11/23/13		
Implement TCVis Controller Interaction Module	21	11/03/13	11/23/13	8	Anthony Alleven
Implement Cursor Tracking Module	21	11/03/13	11/23/13	9	Matt Mayer
Implement JTopenToolKit interaction module	21	11/03/13	11/23/13	10	Logan Scott
Implement IPSI Interaction Module	21	11/03/13	11/23/13	11	James C Erickson
Implement Facade API interaction module	21	11/03/13	11/23/13	12	Paul Uhing
Test Modules	56	10/13/13	12/07/13		
Create VirtuoseAPI Interaction Module Testing Stub	14	10/13/13	10/26/13		James C Erickson
Receive mouse input	14	10/13/13	10/26/13		Anthony Alleven
Receive virtuose input	14	10/13/13	10/26/13		Paul Uhing
Test TCVis module	14	11/24/13	12/07/13	18	Anthony Alleven
Test Cursor Tracking Module	14	11/24/13	12/07/13	19	Matt Mayer

Test JTopenTK module	14	11/24/13	12/07/13	20	Logan Scott
Test IPSI module	14	11/24/13	12/07/13	21	James C Erickson
Test Facade API module	14	11/24/13	12/07/13	22	Paul Uhing
Integration Testing	7	12/08/13	12/14/13		
Test Integration JTopenTk and IPSI modules to describe objects in physics engine	7	12/08/13	12/14/13	29, 30	Logan Scott
Test Integration between IPSI, VirtuoseAPI JTopenTK to perform collision detection	7	12/08/13	12/14/13	29, 30, 31	James C Erickson
Get updated TCVis controller supporting two-way communication	7	01/27/14	02/02/14	Pin	Anthony Alleven
Prepare for mid-year presentation	14	11/25/13	12/08/13		Anthony Alleven
Update for two-way communication	14	02/03/14	02/16/14		
Implement Part selection confirmation from TCVis	14	02/03/14	02/16/14	35	Anthony Alleven
Implement part selection from Virtuose	14	02/03/14	02/16/14	35	Matt Mayer
Update Integration Testing	14	02/03/14	02/16/14	35	Matt Mayer
Make final poster	14	02/17/14	03/02/14	37	James C Erickson
Compile Module Interface document	14	02/17/14	03/02/14	37	Matt Mayer
Make Design Document	14	02/17/14	03/02/14	37	Paul Uhing