MATLAB interface for *cisst* libraries

Group 16
Zachary Zhou
Mentor: Anton Deguetz
Summary

• Goal
  – Create cisst wrapper in MATLAB for ease of access to cisst libraries / data manipulation
  – Automate compilation of library with Cmake
  – Handle passing data from C/MATLAB
Previous Approach

• Compile *cisst* C source code -> MEX files
• Obtain list of functions
• Dynamically generate MATLAB classes to handle *cisst* interface
• Handle sending of data between C/MATLAB
Problems with Old Approach

• MEX issues
  – Requires adding additional code into C source code
    • cisst is stable, should not directly modify class files
  – Only one way to access C code:
    • mexFunction
    • Very limited applications
      – Would require a lot of string manipulation to achieve desired results
New Approach

• Utilizes C Libraries
  – Uses MATLAB’s callLib/loadLibrary functionality

• Wrapping classes:
  – Use pointers/reinterpret_cast
  – Generate MATLAB object in C

• Passing Data
  – Basic types are simple
  – Use wrapped classes to pass composite types
Passing Data from MATLAB/C

• Basic types
  – Int, double, float, String
    • No problem, can be passed as is

– Matrices/arrays
  • Create a MxArray object in C pass to MATLAB
  • Receive as MxArray type, convert into C array
Passing Matrices

• Matrices:
  – Use a MxArray:
    • Have first line (index 0,0) be the type of arguments in String form
    • Remainder of MxArray corresponds directly to C array
  – MATLAB -> C
    • Requires some string manipulation
    • Read in as MxArray
    • Create C array based on argument type
    • Pass contents of MxArray to C array
Passing Composite Types

• Using this approach, we can simply pass composite types (objects) as the pointer to the object in C
• Use reinterpret_cast to retrieve object from MATLAB
• Issue:
  – Error occurs currently, related to the wrapping of classes
  – On hold until class wrapper portion is resolved
Wrapping Classes (C-> Matlab)

- Use a simple C script to wrap classes and pass them to MATLAB
- Script will create an instance of the C object and pass the pointer to the object to MATLAB
- Script will generate MATLAB code (in string form) and pass to MATLAB
  - Utilize the MATLAB evaluate() function to pass code to MATLAB
Current Model of C script

• Object to wrap: ComponentA
  – ComponentA is defined in cisst
  – ComponentA.h/ComponentA.cpp already defined

• C script
  – Generate an object of type Component A
    • Generate object of A, retrieve pointer to the object
String Wrapper(String className, ComponentA* pointer) {
    String[] functionPrototypes = from ComponentA
    String matlabCode =
        "classdef " + className + "
        properties
            Cobject = pointer;
        methods
            /* List of functions */
        end
        end"

    return matlabCode;
}
Current Model (Continued)

• Pseudo code of a function call form matlab (string form)
  – function return_types=function 1{
    float funcPointer= function1 pointer
    callLib("cisstLibraryName","interpret",functionPointer, object pointer, args);
  }

• Interpret function(function pointer, object pointer, args){
  calls function in C on the object using passed arguments
MATLAB side

• Load the library
  – [notfound,warnings]loadlibrary(‘lib.dylib’)
  – String code =Calllib(‘lib’,‘wrapper’,arguments)
  – evaluate (code)

• Utilizing the object in MATLAB
  – Class is already created from calling C method
  – Simply use as follows:
    • ComponentA.function1();
  – Calls C equivalent, and executes on C side
Current Issues

• Because we are using `evaluate(String)` to create an object
  – When we try to create multiple objects of the same class, we get an error in MATLAB:
    • The class is already defined
Solutions

• Add in a separate C script to initialize the object
  – One script for passing class definition to MATLAB
  – One script to check if class def was already passed, if so just call the script to initialize the object
    • Use static types

• Does MATLAB have a class type that can be passed to C?
  – mxArray exists
  – Is there a mxArrayClass or mxArrayStruct to use?
Dependencies

• Find a way to generate 2 instances of same class in MATLAB
  – Error when trying to create 2 instances: class is already defined in matlab, attempts to define twice
Deliverables

• Minimum:
  – Be able to load a single component without configuration file onto MATLAB
  – Get dynamic loading to work
  – Write basic data conversion methods for native types

• Expected:
  – Utilize CMake to built MATLAB plug-in library
  – Create MATLAB object on the fly with string names
  – Populate MATLAB with component interfaces, names, and commands
  – Conversion methods for vectors and matrices
  – Proper documentation of completed portions

• Maximum:
  – Conversion methods for composite types (cisstDataGenerator)
  – Test on multiple machines from MATLAB
  – Try running MATLAB wrapper from command-line
  – Extensive documentation/readme
Milestones

• Explore C/MATLAB interfaces
  – Complete by: March 1st
  – Status: in progress
• Dynamic loading working on cisst
  – Complete by: April 15th
• Data Conversion (basic)
  – Completed April 6th
• Data Conversion (composite)
  – Complete by: April 15th
• Use CMake to build plugin library
  – Completed
• Composite objects and populate MATLABInterface with interface names/components
  – Complete by: May 10th
• Documentation:
  – Complete by: May 10th
## Timeline

<table>
<thead>
<tr>
<th>Deliverables</th>
<th>20-Feb</th>
<th>1-Mar</th>
<th>9-Mar</th>
<th>16-Mar</th>
<th>23-Mar</th>
<th>2-Apr</th>
<th>6-Apr</th>
<th>13-Apr</th>
<th>20-Apr</th>
<th>27-Apr</th>
<th>4-May</th>
<th>10-May</th>
</tr>
</thead>
<tbody>
<tr>
<td>Read/understand cisst library</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Explore MATLAB/C interfaces</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Call a C method from MATLAB</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Call MATLAB from C</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Pass Variables between C/MATLAB</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Build plugin library</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Load single component on MATLAB</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Conversion of Basic Data Types</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Conversion of user defined types (cisstDataGenerator)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Software Documentation</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Final Report</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**In progress**

**Completed**
References

• https://trac.lcsr.jhu.edu/cisst
• https://trac.lcsr.jhu.edu/cisst/wiki/cisstMultiTaskTutorial
• http://www.mathworks.com/support/tech-notes/1600/1605.html
• http://www.cmake.org/cmake/resources/resources.html
Thank you

Questions?