ERC | CISST

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/loadLibary 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

- Matricies:
  - 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
  - Ultilize 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

Current Model (Continued)

```
String Wrapper(String className, ComponentA* pointer){
        String[] functionPrototypes= from ComponentA
        String matlabCode=
                "classdef "+className+"/n
                        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]loadlibary(‘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 mxClass or mxStruct 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

ERC | CISST

Timeline

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

🟨 In progress

🟦 Complete

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?