Artificial Pancreas Simulator

PRESENTED BY ANAS EL FATHI

Objective

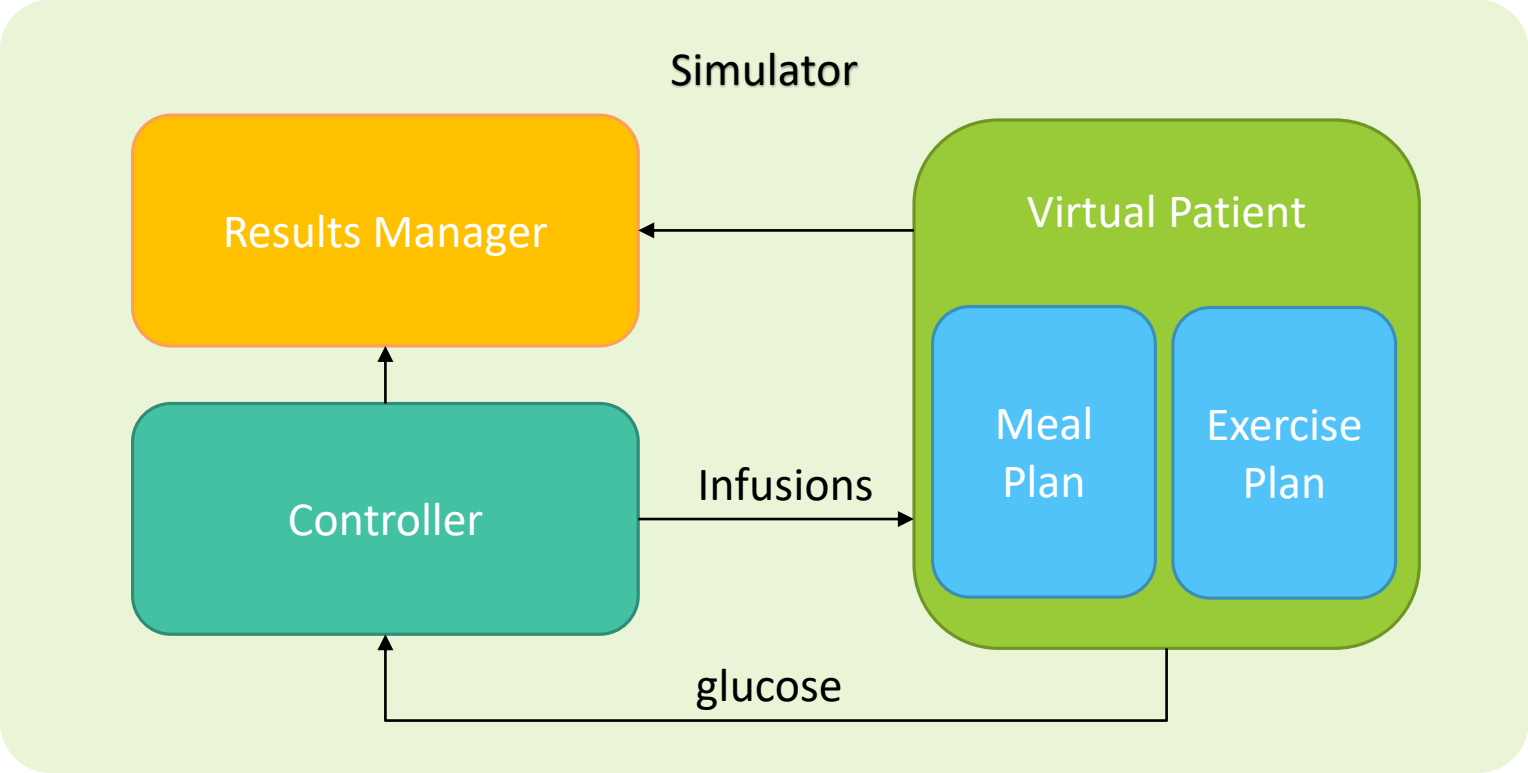
- Simulate a T1D patient:
  - Model Insulin, meals, exercise effects on glucose.
  - Model T1D patients individual variability.
  - Model patient day-to-day variability.

- Test Artificial Pancreas systems:
  - Prototype novel controllers.
  - Evaluate performances and fine-tune algorithms.

Characteristics

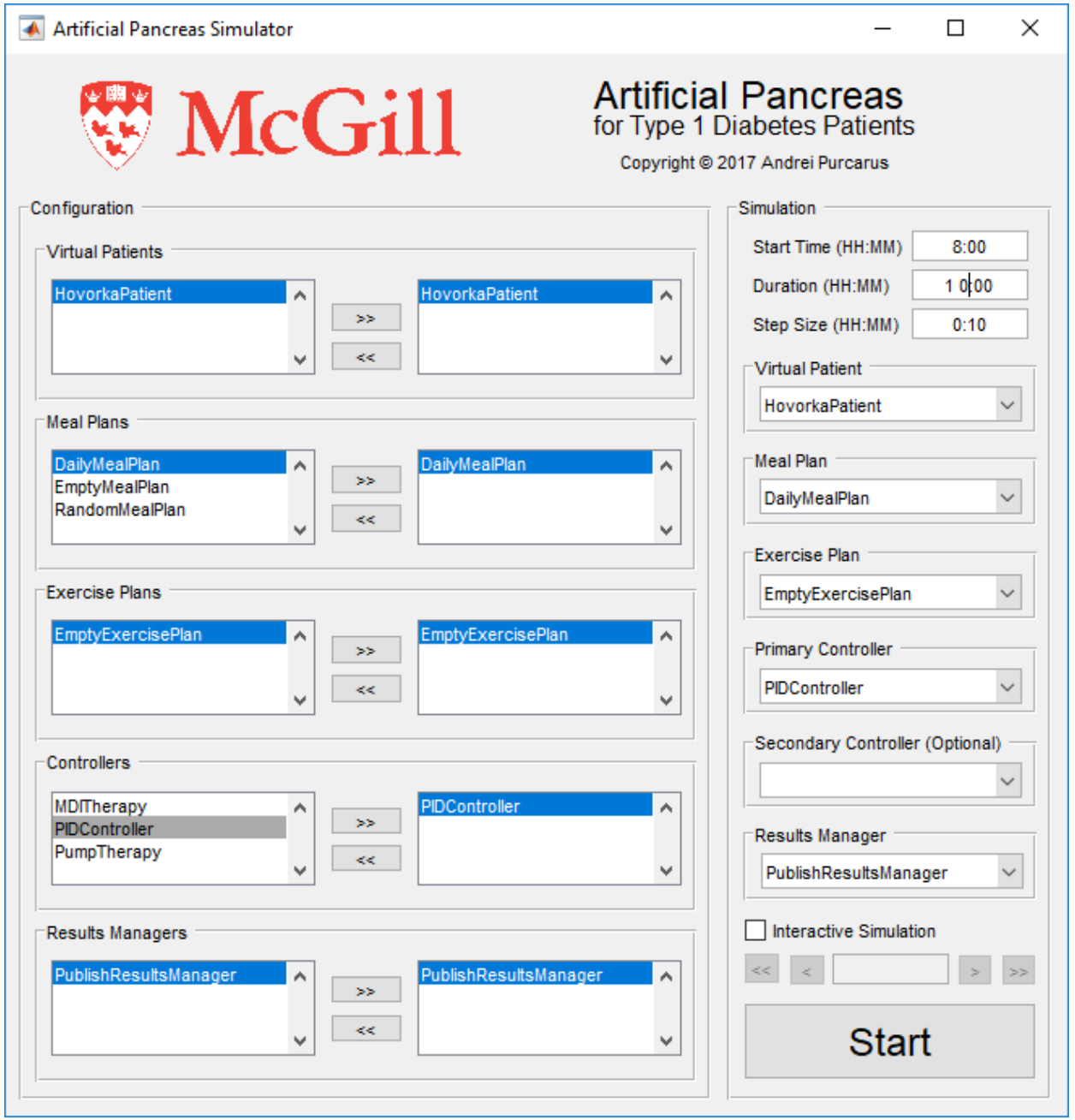
- A modular, open-ended-design.
  - Easily manipulated framework, but not necessarily a complete solution.
  - has a minimal working setup, but support a lot of configuration for the full experience.
- Usable by expert and non-expert users.
  - GUI vs command line.

Architecture



Configuration

- Simulation Start Time
- Simulation Duration
- Simulation Step Size
- Virtual Patients:
  - patient model
  - meal plan
  - exercise plan
  - controller
- Result Manager


Artificial Pancreas Simulator
McGill
Artificial Pancreas
for Type 1 Diabetes Patients
Copyright © 2017 Andrei Purcarus
Configuration
Virtual Patients
HovorkaPatient
>>
<<
HovorkaPatient
Meal Plans
DailyMealPlan
EmptyMealPlan
RandomMealPlan
>>
<<
DailyMealPlan
Exercise Plans
EmptyExercisePlan
>>
<<
EmptyExercisePlan
Controllers
MDITherapy
PIDController
PumpTherapy
>>
<<
PIDController
Results Managers
PublishResultsManager
>>
<<
PublishResultsManager
Simulation
Start Time (HH:MM)
8:00
Duration (HH:MM)
1 0:00
Step Size (HH:MM)
0:10
Virtual Patient
HovorkaPatient
Meal Plan
DailyMealPlan
Exercise Plan
EmptyExercisePlan
Primary Controller
PIDController
Secondary Controller (Optional)
Results Manager
PublishResultsManager
Interactive Simulation
<<
<
>
>>
Start

GUI

Process

- For each time: from start until start + duration
  - Call the controller: getInfusions.
  - Call the patient model: updateState.
- At the end: displayResults.

VirtualPatient

```
classdef VirtualPatientTemplate < VirtualPatient

    properties(GetAccess = public, SetAccess = private)
        % Declare any internal variables used by the class here.
        n = 10; % Example: the number of elements to hold in the state vector.
        param; % Example: a struct containing the patient model parameters.
        state; % Example: an n x 1 column vector containing the internal patient state.
    end

    methods
        function this = VirtualPatientTemplate(mealPlan, exercisePlan, options)
            % Default constructor

            % Calls the base class constructor to initialize the base object. This is required.
            this@VirtualPatient(mealPlan, exercisePlan);

            % Write your code to initialize the virtual patient here.
            this.param = struct(); % Example: initialize the model parameters to an empty struct.
            this.param.name = options.name; %Example: parameters has a name field which is initialized from options.
            this.state = zeros(this.n, 1); % Example: initialize the state to a column vector of zeros.
        end

        function prop = getProperties(this)
            prop = this.param; % Example: return the internal model parameter struct.
        end

        function meas = getGlucoseMeasurement(this)
            meas = this.state(end); % Example: the last element in the state vector could store the glucose level.
        end

        function updateState(this, startTime, endTime, infusions)
            % Write your code to update the state of the patient here.

            Ubasal = infusions.basalInsulin; % Example: get the current basal insulin infusions.
```

```
classdef VirtualPatientTemplate < VirtualPatient

    properties(GetAccess = public, SetAccess = private)
        % Declare any internal variables used by the class here.
        n = 10; % Example: the number of elements to hold in the state vector.
        param; % Example: a struct containing the patient model parameters.
        state; % Example: an n x 1 column vector containing the internal patient state.
    end

    methods
        function this = VirtualPatientTemplate(mealPlan, exercisePlan, options)
            % Default constructor

            % Calls the base class constructor to initialize the base object. This is required.
            this@VirtualPatient(mealPlan, exercisePlan);

            % Write your code to initialize the virtual patient here.
            this.param = struct(); % Example: initialize the model parameters to an empty struct.
            this.param.name = options.name; %Example: parameters has a name field which is initialized from options.
            this.state = zeros(this.n, 1); % Example: initialize the state to a column vector of zeros.
        end

        function prop = getProperties(this)
            prop = this.param; % Example: return the internal model parameter struct.
        end

        function meas = getGlucoseMeasurement(this)
            meas = this.state(end); % Example: the last element in the state vector could store the glucose level.
        end

        function updateState(this, startTime, endTime, infusions)
            % Write your code to update the state of the patient here.

            Ubasal = infusions.basalInsulin; % Example: get the current basal insulin infusions.
            Ubolus = infusions.bolusInsulin; % Example: get the current bolus insulin infusions.

            meal = this.mealPlan.getMeal(startTime); % Example: get the current meal.
            exercise = this.exercisePlan.getExercise(startTime); % Example: get the current exercise.

            % Write the differential equations relating the patient state to the infusions/meal/exercise here.
```

VirtualPatient

```
properties(GetAccess = public, SetAccess = private)
    % Declare any internal variables used by the class here.
    n = 10; % Example: the number of elements to hold in the state vector.
    param; % Example: a struct containing the patient model parameters.
    state; % Example: an n x 1 column vector containing the internal patient state.
end

methods
    function this = VirtualPatientTemplate(mealPlan, exercisePlan, options)
        % Default constructor

        % Calls the base class constructor to initialize the base object. This is required.
        this@VirtualPatient(mealPlan, exercisePlan);

        % Write your code to initialize the virtual patient here.
        this.param = struct(); % Example: initialize the model parameters to an empty struct.
        this.param.name = options.name; %Example: parameters has a name field which is initialized from options.
        this.state = zeros(this.n, 1); % Example: initialize the state to a column vector of zeros.
    end

    function prop = getProperties(this)
        prop = this.param; % Example: return the internal model parameter struct.
    end

    function meas = getGlucoseMeasurement(this)
        meas = this.state(end); % Example: the last element in the state vector could store the glucose level.
    end

    function updateState(this, startTime, endTime, infusions)
        % Write your code to update the state of the patient here.

        Ubasal = infusions.basalInsulin; % Example: get the current basal insulin infusions.
        Ubolus = infusions.bolusInsulin; % Example: get the current bolus insulin infusions.

        meal = this.mealPlan.getMeal(startTime); % Example: get the current meal.
        exercise = this.exercisePlan.getExercise(startTime); % Example: get the current exercise.

        % Write the differential equations relating the patient state to the infusions/meal/exercise here.
        this.state = ones(this.n, 1); % Example: update the state.
    end
end

methods(Access = private)
    % Define any additional internal functions here. These functions
    % cannot be accessed from outside of this class.

end
```

VirtualPatient

```
    methods
        function this = VirtualPatientTemplate(mealPlan, exercisePlan, options)
            % Default constructor

            % Calls the base class constructor to initialize the base object. This is required.
            this@VirtualPatient(mealPlan, exercisePlan);

            % Write your code to initialize the virtual patient here.
            this.param = struct(); % Example: initialize the model parameters to an empty struct.
            this.param.name = options.name; %Example: parameters has a name field which is initialized from options.
            this.state = zeros(this.n, 1); % Example: initialize the state to a column vector of zeros.
        end

        function prop = getProperties(this)
            prop = this.param; % Example: return the internal model parameter struct.
        end

        function meas = getGlucoseMeasurement(this)
            meas = this.state(end); % Example: the last element in the state vector could store the glucose level.
        end

        function updateState(this, startTime, endTime, infusions)
            % Write your code to update the state of the patient here.

            Ubasal = infusions.basalInsulin; % Example: get the current basal insulin infusions.
            Ubolus = infusions.bolusInsulin; % Example: get the current bolus insulin infusions.

            meal = this.mealPlan.getMeal(startTime); % Example: get the current meal.
            exercise = this.exercisePlan.getExercise(startTime); % Example: get the current exercise.

            % Write the differential equations relating the patient state to the infusions/meal/exercise here.
            this.state = ones(this.n, 1); % Example: update the state.
        end
    end

    methods(Access = private)
        % Define any additional internal functions here. These functions
        % cannot be accessed from outside of this class.

    end

end
```

VirtualPatient

```
        % Write your code to initialize the virtual patient here.
        this.param = struct(); % Example: initialize the model parameters to an empty struct.
        this.param.name = options.name; %Example: parameters has a name field which is initialized from options.
        this.state = zeros(this.n, 1); % Example: initialize the state to a column vector of zeros.
    end

    function prop = getProperties(this)
        prop = this.param; % Example: return the internal model parameter struct.
    end

    function meas = getGlucoseMeasurement(this)
        meas = this.state(end); % Example: the last element in the state vector could store the glucose level.
    end

    function updateState(this, startTime, endTime, infusions)
        % Write your code to update the state of the patient here.

        Ubasal = infusions.basalInsulin; % Example: get the current basal insulin infusions.
        Ubolus = infusions.bolusInsulin; % Example: get the current bolus insulin infusions.

        meal = this.mealPlan.getMeal(startTime); % Example: get the current meal.
        exercise = this.exercisePlan.getExercise(startTime); % Example: get the current exercise.

        % Write the differential equations relating the patient state to the infusions/meal/exercise here.
        this.state = ones(this.n, 1); % Example: update the state.
    end
end

methods(Access = private)
    % Define any additional internal functions here. These functions
    % cannot be accessed from outside of this class.

end

end
```

VirtualPatient

```
classdef InfusionControllerTemplate < InfusionController

    properties (GetAccess = public, SetAccess = private)
        % Declare any internal variables used by the class here.

    end

    methods
        function this = InfusionControllerTemplate(simulationDuration, simulationStartTime, simulationStepSize, patient, options)
            % Default constructor

            % Calls the base class constructor to initialize the base object. This is required.
            this@InfusionController(simulationDuration, simulationStartTime, simulationStepSize, patient);

            % Write your code to initialize the infusion controller here.

        end

        function infusions = getInfusions(this, currentTime)
            % Define how other entities (such as the virtual patient) get the current infusions here.
            infusions.basalInsulin = 0; % Example: set the basal insulin.
            infusions.bolusInsulin = round(this.patient.getMeal(currentTime).value / 10); % Example: set the bolus insulin.
        end
    end

    methods (Access = private)
        % Define any additional internal functions here. These functions
        % cannot be accessed from outside of this class.

    end

end
```

PrimaryController