Stochastic Optimisation Algorithms Test Bed

- Applicble for bench marking both Model-Based and Model-Free approaches

class(Control_Coordinator)

Description (high level):

This object is wraps together the full control module for the Black-Box Stochastic Optimisation Test-Bed. This object has only two functions:

"initialiseControllers" - this function initialises the controller object that actually perform the computation for the control action to be applied at each time step.

"computeControlAction" - this function compute the control action at each time step based on the latest information. It is explained more fully in the simulator wireframe

In the initialise function, the number and structure of the controller objects is be defined. The wireframe below describes how this strucuture is built.

Properties

Private:

- classNameLocal	The name of the class for the Local Controller, this MUST inherit from the Abstract Class called "Control_LocalController"
- classNameGlobal	The name of the class for the Global Controllers, this MUST inherit from the Abstract Class called "Control_GlobalController"
- globalController	The object that is the Global Controller
- localControllerArray	The Array of objects that are the Local Controllers
- modelType	The type of system that is being controlled (e.g. a "building")
- vararginLocal	The variable arguments that can be set in "main.m" and are passed directly to the initialisation of each Local Controller
- vararginGlobal	The variable arguments that can be set in "main.m" and are passed directly to the initialisation of each Local Controller
- stateDef	The State Definition object - each "Control_Coordinator" has a copy of the State Defintion object so that the number of local controllers and their masks can be changed separate from the other "Control_Coordinator"s
- constraintDef	The constraint Definition object
- numControllers	The number of local controllers (this should always equal "length(localControllerArray)" and so it technically a redundant property

Functions:

Public:

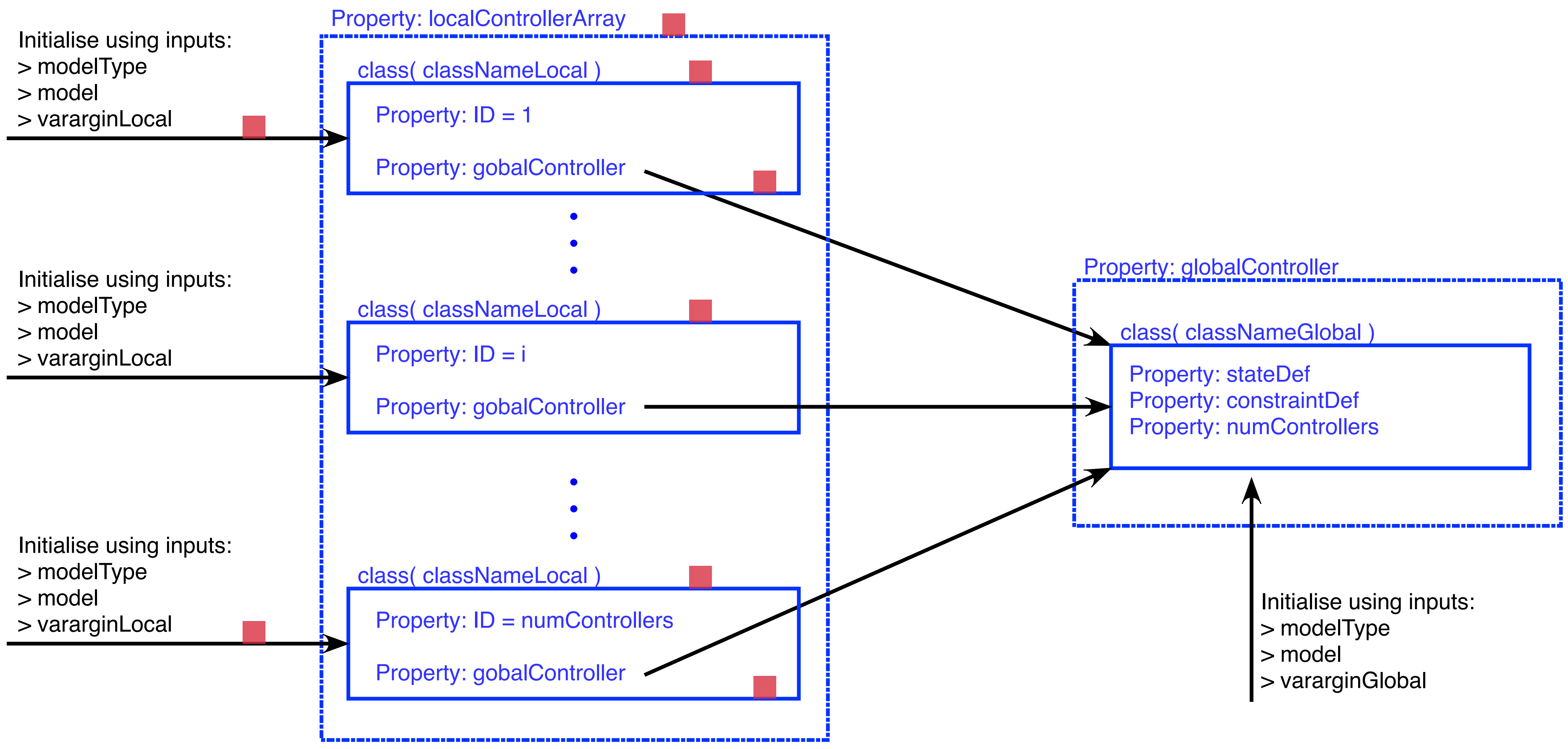
- [ ] = initialiseControllers(obj , inputSettings , inputModel)
- [u , diagnostics] = computeControlAction(obj , x , xi , stageCost , prediciton)

FUNCTION: [ ] = initialiseControllers(obj , inputSettings , inputModel)

NOTE: "inputSettings" has a property call "globalInit" that stems directly from the property of the same name sepcified in "main.m" for this control technique

NOTE: "inputSettings" has a property call "modelFree" that stems directly from the property of the same name sepcified in "main.m" for this control technique. The model of the system is passed to "initialise_..." functions mentioned below when this property is "false"

```
if inputSettings.globalInit
    > A single Global Controller object of "classNameGlobal" is created with properties of the full State Definition object and Constraint Definiton object
    > The "[...] = initialise_globalControl(...)" function is call on this single Global Controller object that was created
        > This "initialise_globalControl" needs to be updated if something other than the default control structure is desired
        > For example if implmenting centralised control control, then this "initialise_globalControl" should return that there is only 1 local controller and that this controller has access to all the information

else
    > There is no Global Controller object created and the default control structure specified by the system will be used for creating the local controllers

end


for iController = 1: obj.numControllers
    > A Local Controler object of "classNameLocal" is created with properties:
        > "iController" to identify which local controller it is
        > A "partial" State Definiton object that specifies the variable dimensions for this controller
        > A "partial" Constraint Definiton object that specifies contraints relative to the "partial" State Definiton object
            > The "partial" Constraint Definiton object includes any polytopic constraint which are involved variables relevant to this controller
        > A handle to be able to access the single Global Controller object (i.e. "globalController")

    > The "[...] = initialise_localControl(...)" function is call on this Local Controller object
        > This "initialise_localControl" should be used to perform any offline computation to speed-up controller during the simulation run

end
```



Property: localControllerArray
Initialise using inputs:
> modelType
> model
> vararginLocal
class(classNameLocal)
Property: ID = 1
Property: gobalController
Initialise using inputs:
> modelType
> model
> vararginLocal
class(classNameLocal)
Property: ID = i
Property: gobalController
Initialise using inputs:
> modelType
> model
> vararginLocal
class(classNameLocal)
Property: ID = numControllers
Property: gobalController
Property: globalController
class(classNameGlobal)
Property: stateDef
Property: constraintDef
Property: numControllers
Initialise using inputs:
> modelType
> model
> vararginGlobal