Real-Time Simulation of CESA-I Central Receiver Solar Thermal Power Plant

Javier Bonilla[1] Lidia Roca[1] Luis J. Yebra[1] Sebastián Dormido[2]
[1]CIEMAT- Plataforma Solar de Almería, Ctra. Senes s/n, 04200 Tabernas, Almería, Spain
Centro de Investigaciones Energéticas, MedioAmbientales y Tecnológicas
[2]Universidad Nacional de Educación a Distancia (UNED), Madrid, Spain

Abstract

This paper presents a real-time heliostat field simulator, based on a hybrid model, using Modelica as the modelling language. The development of industrial dynamic simulators, in this work for a central receiver solar thermal power plant: CESA-I from CIEMAT-PSA, is mainly aimed as a tool for the enhancement of advanced control algorithms but it is also useful for training purposes. The developed real-time heliostat field simulator is basically the union of the hybrid heliostat field model and a wrapped model which handles the real-time simulation and communication issues between the heliostat field simulator and HelFiCo (Heliostat Field Control) software which is in charge of manipulating and controlling the heliostat field according to an automatic control strategy. The real-time heliostat field simulator provides a virtual system, with the same response as the real plant.

Keywords: real-time simulation; hybrid modelling; heliostat field; solar thermal power plant

1 Introduction

Design and development of dynamic models for simulation and control system design purposes, is gaining importance in solar thermal industrial processes. An example is the deployment of advanced control systems that optimize the overall performance of solar thermal power plants.

It is a fact nowadays that this task is a priority research line [13] at CIEMAT National Spanish Laboratories (Centro de Investigaciones Energéticas, Medioambientales y Tecnológicas - Research Centre for Energy, Environment and Technology), public organization owned by the Spanish Ministry of Science and Innovation.

During the last years a big effort has been devoted to the development of control systems for solar thermal power plants,making an important part of the experiences directly against the real plant. These real tests have increased the resources needed for development. Nevertheless, some of these systems, are expensive, scarce and present a costly experimentation time. This fact has motivated the development of a dynamic model for the CESA-I heliostat field plant, aimed mainly as a tool for the enhancement of advanced control algorithms. A preliminary research work in this topic was published in [1].

Several softwares have been developed to calculate the solar-flux density distribution on a central receiver system [6] with the aim of optimizing the components of the receiver, studying the optical performance and improving the process. These algorithms include geometry which depends on: the sun vector, the heliostats and tower positions, properties of the components (such as the reflectivity of the mirrors), errors (such as wrong canting), shadows, atmospheric attenuation, etc.

This paper explains the heliostat dynamic as a function of the messages received from a heliostat field control, although a future goal in our working is to include a simplified optical model to calculate the flux distribution caused by a heliostat.

2 Goals

The main goal of this paper is to model and develop a customizable real-time central receiver thermal power plant simulator, which is adapted to CESA-I test-bed facility requirements.

This development is mainly aimed as a tool for the enhancement of advanced control algorithms and also for training purposes. Design, testing and

DOI: 10.3384/ecp09430062

validation of new advanced control strategies in real plants, is expensive, dangerous and requires a long testing time. It is intended to weed out such problems by an efficient, safe and reliable real-time simulator.

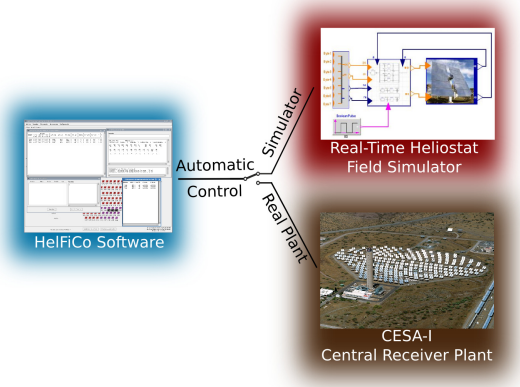

Figure 1: Relationship between HelFiCo software, the real-time heliostat field simulator and CESA-I central receiver plant.

The real-time simulator has to provide a virtual system, with the same behaviour and response as the real plant (see Figure 1). The communication between HelFiCo (Heliostat field control) software and the real-time simulator has to be transparent for the user of the heliostat field control software.

3 Central Receiver Solar Thermal Power Plants

In this section an overview of the basic components and operating procedures for a Central Receiver Solar Thermal Power Plant (CRSTPP) are introduced. Figure 2 shows an explicative diagram of a general CRSTPP.

The operation of this kind of plants is based on the concentration of incoming solar energy using a heliostat field that reflects the incident solar radiation onto a (typically volumetric) receiver (theoretically onto an optical point in the 3-D space). As the sun position changes during the day, each heliostat of the field has to change its position in real time according to the selected aiming point on the receiver, as different aiming points can be selected in order to achieve a uniform temperature distribution on the receiver [5]. The receiver is located at the top of the tower (84m height in CESA-I) and acts as an energy exchanger, receiving solar energy and transferring it to a thermo-hydraulic circuit with air medium (see Figure 2). The system is also composed of an energy storage tank, an air/water-steam heat exchanger (evaporator), blowers and valves. The combined action of the blowers allows feeding either the storage tank or the heat exchanger with hot air. The evaporator is formed of the primary circuit and a secondary one with sub-cooled inlet water and with superheated steam outlet. A measurement of the overall concentrated input radiation, a controlled water pump and an outlet controlled valve define the main boundary conditions for the system.

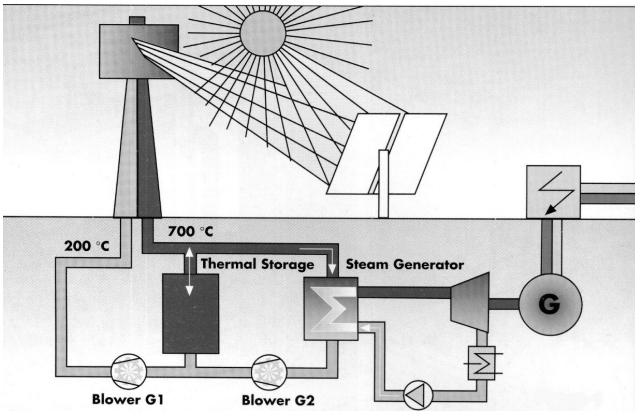

Figure 2: Schematic diagram of a central receiver solar thermal power plant [13].

3.1 CESA-I test-bed facility

The modelling and simulation activities object of this work are focused on the CESA-I facility, a central receiver solar thermal power plant belonging to CIEMAT, and located at the Plataforma Solar de Almería (PSA), South-East of Spain. This test-bed plant can be seen in Figure 3, and it is an experimental prototype for electricity generation among other research projects.

The CESA-I facility collects direct solar radiation by means of a field of 300 heliostats (39.6-m^2-surface) distributed in a 330-x-250-m north field into 16 rows. The heliostats (see Figure 4) have a nominal reflectivity of 92%, the solar tracking error on each axis is 1,2 mrad and the reflected beam image quality is 3 mrad. North of the heliostat field there are two additional test zones for new heliostat prototypes, one located 380 m from the tower and the other 500 m away. The maximum thermal power delivered by the field onto the receiver aperture is 7 MW. At a typical de-

sign irradiance of 950 W/m2, a peak flux of 3.3 MW/m2 is obtained. In addition, the 99% of the power is focused on a 4m-diameter circle, 90% in a 2.8-m circle.

Figure 3: CESA-I facility (CIEMAT-PSA).

The 80-m-high concrete tower has a 100ton load capacity. The tower is complete with a 5-ton-capacity crane at the top and a freight elevator that can handle up to 1000-kg loads. For those tests that require electricity production, the facility has a 1.2 MW two-stage turbine in a Rankine cycle designed to operate at 520 Â°C 100 bar superheated steam.

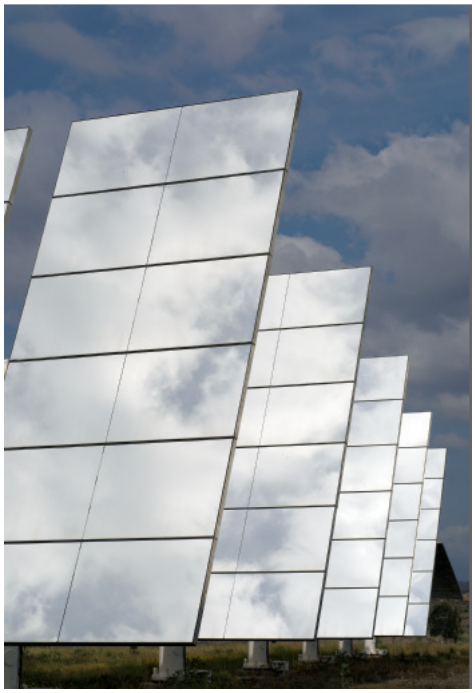

Figure 4: Heliostats from CESA-I facility.

4 Heliostat Field Modelica Model

To take advantage of all the Modelica features, the heliostat field simulator has been developed using this language. Modelica is a standard unified modelling language [8] with many advantages for modelling dynamic systems, because it is both, an object-oriented and acausal language. The object-oriented feature allows developing a set of reusable objects which can be used in future developments and the acausal feature allows describing the behaviour of dynamic systems using the differential equation systems which describe them. Dynamic behaviour and numerical aspects are taken into account in Modelica, because it provides equation sections and event modelling [4].

Moreover, Modelica has a library, *StateGraph*, for modelling hierarchical state machines. The *StateGraph* Modelica library offers features to define conveniently discrete events and reactive systems in Modelica models. Since Modelica is used as an action language, a Modelica translator can guarantee that a State-Graph has deterministic behaviour. StateGraph models can be combined with components of any other Modelica library and can therefore be very easily used to control a continuous plant [9].

A simple hybrid model of a single heliostat has been developed combining the system dynamic continuous variables with operation mode discrete variables. This one-heliostat model may be briefly explained as follows:

- Movement dynamic. The continuous variables of the system are the azimuth, a, and elevation positions, e, which can be obtained through movement equations using the angular velocities as known parameters (ω_a, ω_e).

$$\frac{da}{dt} = \omega_a; \quad \frac{de}{dt} = \omega_e$$

- Operation mode. Not only each single heliostat must achieve the desired reference position defined by the control system software, but also communication failures and timeouts must be taken into account by the heliostat. These characteristics have been included in a state machine using the *StateGraph* library.

  Five main operation modes are defined:

- *Waiting*: The heliostat is waiting for an input message. In this mode, the heliostat may be moving to a position.
- *Request*: The heliostat sends an output message including position information.
- *Reset*: When the heliostat is in reset mode, that updates its positions moving to zero position for azimuth and elevation, moving firstly in the azimuth direction. Once it get to zero-azimuth position, it begins moving in the elevation axis. In this operation mode, the heliostat does not send any output message until the reset time, *Treset*, is reached.
- *Stow*: If the heliostat does not receive an input message during *Timeout* seconds, it moves to stow (zero) position. In this case, the movement is in the two axes at the same time.
- *Control*: The heliostat moves to the position reference specified in the input message.
- *Request + Control*: Request and Control modes at the same time.
- *Request + Reset*: Request and Reset modes at the same time.

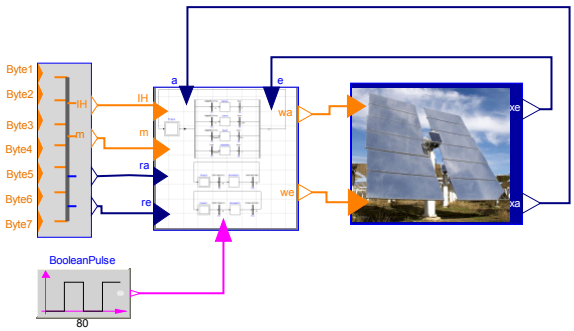

Figure 5: The hybrid one-heliostat model

Figure 5 shows the one-heliostat model which has the following features:

- The input signals are a collection of integer signals which represent the received bytes from the heliostat field control software. Notice that these input signals are the same for all the heliostats which belong to the same heliostat row.
- These bytes are decoded to obtain the heliostat identifier (*IH*) parameter (which identifies univocally each heliostat), the control message (*m*) and the azimuth and elevation references (*ra*, *re*).
- A state machine component (see Figure 6), based on the *StateGraph* library, makes possible to know the angular velocities in both axes (*wa*, *we*) which depend on the heliostat state.

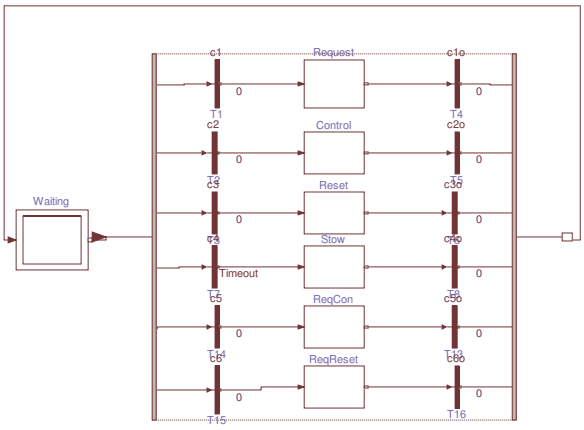

Figure 6: The state machine diagram

- The azimuth and elevation positions (*a*, *e*) are calculated with the movement dynamic using the angular velocities provided by the state machine component.

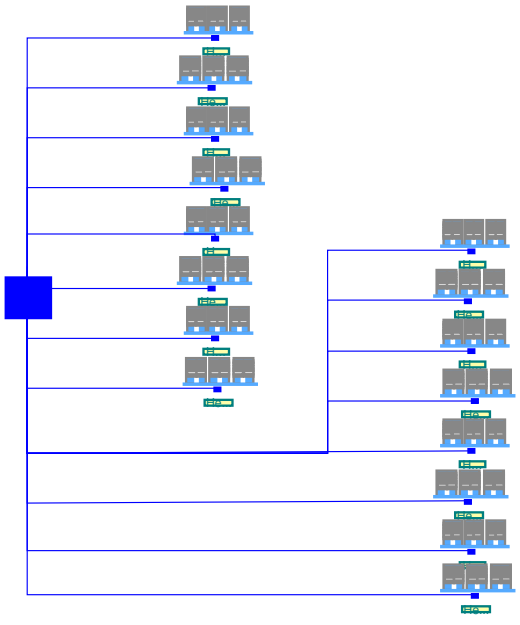

Figure 7: Heliostat field model

A heliostat row model is $NRow_i$ heliostat model instantiations, being $NRow_i$ a parameter that defines the number of heliostats in the i row. Notice that the control system software sends input commands to each one of the 16 rows, therefore the heliostat field simulator is composed of 16-heliostat-row instantiations (see Figure 7).

4.1 Simulation results

In this section a simulation of 16-heliostat row is explained. The inputs messages are shown in Table 1. The heliostat identifiers (IH) are $IH = \{1,2,3,...,15,16\}$, just the heliostat which identifier matches with the one included in the input messages, must process it.

Table 1: Simulation results. Input messages.

Time (s)	IH	Message	ra	re
$t_0 = 0$	10	Request+Reset	100	100
$t_1 = 82$	10	Request+Reset	100	100
$t_2 = 162$	10	Request	0	0
$t_3 = 242$	10	Request	0	0
$t_4 = 322$	10	Request	0	0
$t_5 = 402$	3	Request+Control	100	100
$t_6 = 482$	3	Request+Control	100	100

The output message (see Table 2) specifies the heliostat identifier which has to respond to the request, including its current position, (a, e), and a boolean, *ref*, that turns to 1 when the heliostat achieves the desired reference. At the beginning of the simulation, the heliostat 10 moves to zero position because of the reset message. At time t_2 the heliostat has reached the zero value in azimuth and the output value in this direction is reestablished. The same situation occurs at time t_3 with the elevation direction. The last two messages order the heliostat 3 to move to (100,100) coordinates with a movement request. The output heliostat message at time t_5 shows that its position is (0,0) whereas at time t_6 is (100,100), so the desired reference is reached.

The position of the two heliostats during the simulation are shown in Figure 8 together with the angular velocities in each direction. These angular velocities are parameters that can be changed depending on the heliostat features. For this simulation the velocities were chosen to be ± 5 positions/s. On the other hand, the operation modes for both heliostats are shown in Figure 9.

Table 2: Simulation results. Output messages.

Time	IH	a	e	ref
t_0	10	600	600	0
t_1	10	190	600	0
t_2	10	0	394	0
t_3	10	0	0	0
t_4	10	0	0	0
t_5	3	0	0	0
t_6	3	100	100	1

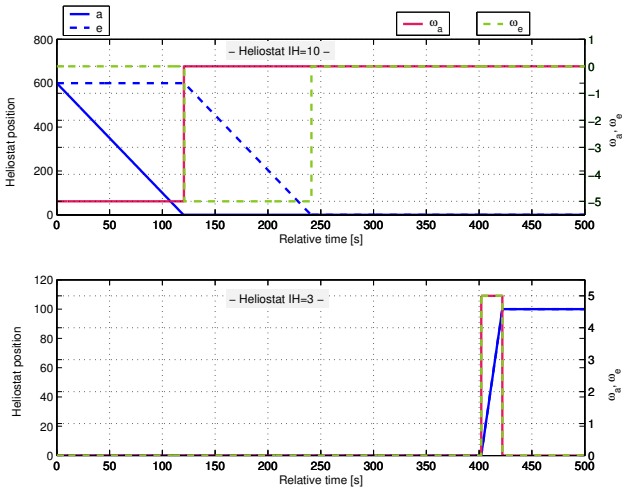

Figure 8: Simulation results. Heliostat positions.

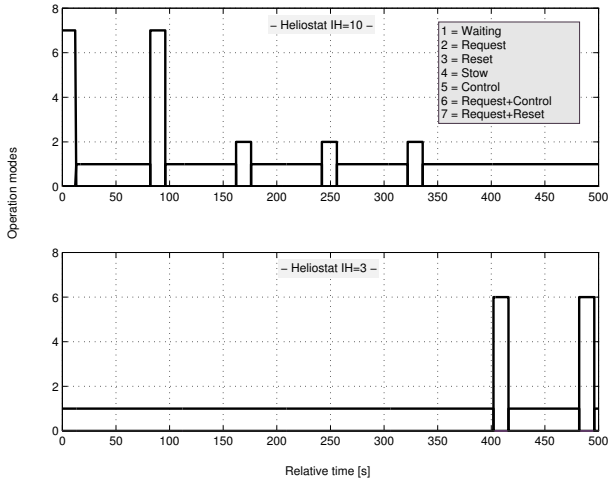

Figure 9: Simulation results. Operation modes.

5 Real-Time Heliostat Field Simulator

The developed real-time heliostat field simulator is mainly the union of the hybrid heliostat field model and a wrapped model which handles the real-time simulation and communication issues. The heliostat field simulator is communi-

cated with HelFiCo (Heliostat Field Control) software, a heliostat field control software [7] which is in charge of manipulating and controlling the heliostat field according to an automatic control strategy.

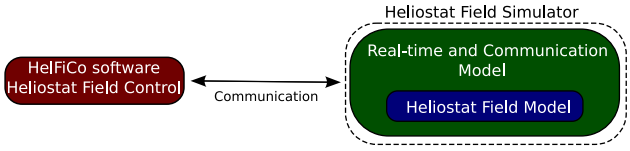

Figure 10: Heliostat field simulator and HelFiCo software block diagram

The figure 10 shows a block diagram which illustrates the concept previously explained and shows the interconnection between the heliostat field simulator and HelFiCo software.

5.1 Real-time and Communication Model

The real-time and communication model wraps the main model, the heliostat field model. This wrapped model provides the following functionality:

- *Real-time simulation:* Synchronize the simulation time dynamically to real-time.
- *Communication issues:* Receive input commands from the heliostat field control software, decode input commands, encode output data from the heliostat field model and send output data to the real-time simulator.

5.1.1 Real-time Model

Real-time demands that the simulation time must be shorter than the simulated time. Since this condition is a fundamental requirement for models to be used in real-time simulations, in some cases, the complexity of the model must be reduced at the expense of loosing quality.

In our case it was not necessary to reduce the complexity of the heliostat field model, just take into account some improvement in the model to optimize the computational efficiency. These improvement will be discussed in the subsection 5.1.2.

Modelica does not have synchronisation support, some Modelica IDEs have this kind of support but it is specific to the IDE tool, such as Dymola [2]. Moreover the synchronisations support in Dymola has the drawback that simulation speed is not limited if simulation time is behind real world time, this can lead to undesirable behaviour of the simulation.

There is an open source real time option for Dymola called JPAARealTime [3]. Since JPAARealTime makes use of Windows libraries it can only be run on Windows operating system. But it was required a multi-platform library for future development in different operating systems. For these reasons a real-time library has been developed for this task. The developed RTSS (Real-Time SimulationS) library allows real-time synchronisation support.

The RTSS library samples the simulation time at certain time instants and stops the current simulation until the simulation time reaches the real time according to the computer's time when the simulation started. This concept is shown in Figure 11.

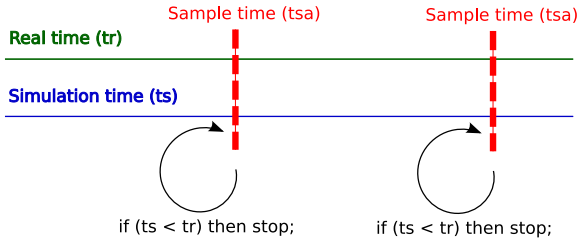

Figure 11: Real-time synchronisation concept

5.1.2 Communication Model

The communication model simulates the behaviour of the communication line between the simulator and HelFiCo (Heliostat Field Control) software, involving receive, send, encode and decode messages. Message decoding is the process of translating the incoming binary message from the heliostat field control software into the corresponding input variables in the Modelica model. On the other hand, message encoding is the reverse process.

The real communication channels between HelFiCo software and CESA-I test-bed facility are RS-485 and RS-232 wires. Each one of the 16 heliostat rows has a shared RS-485 wire, all of them converge to a RS-485 to RS-232 interface converter which is directly connected with the computer where HelFico software is running.

The communication line between both computer programs is simulated by two FIFO (First In, First Out) files in the operative system for each heliostat row in the field, one for sending and the other one for receiving. This FIFO files are share by HelFico software and the real-time simulator, but the operation mode is different in each one. For each heliostat row, the HelFico software open one FIFO file in writing mode (output line) and the other one in reading mode (input line). In the real-time simulator the operation mode of the FIFO files is the opposite, this behaviour is illustrated in Figure 12.

FIFO files were chosen as the inter process communication method for serveral reason. First of all, the FIFO files behaviour is suitable for the developed simulator. The simulator requires just message passing, in order of arrival, as in a queue, and this can be easily obtained using FIFO files, moreover it is not needed synchronization, shared memory or remote procedure calls. Another reason is that the FIFO files behaviour is quite similar to the RS-485 and RS-232 wires behaviour. There are four main operations: open, send, receive and close. It is straightforward to develop an abstract parent class which set the common interface with these four main operations. Then, two subclasses are more specialized versions, inherit attributes and behaviours from the parent class, and introduce their own behaviour. One of these classes implements the FIFO files communication and the other one controls the real communication channels. These two classes allow HelFiCo software to switch between them to communicate properly to the simulator or the real plant. New subclasses can be developed for different kinds of communication, the only requirement is to inherit from the abstract parent class.

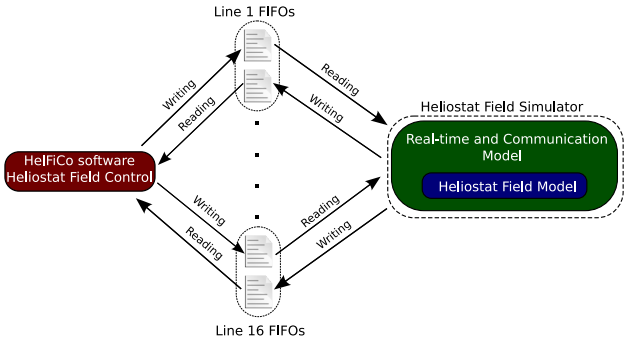

Figure 12: Communication model

5.1.3 Improving the real-time simulator performance

- *Coupling the real-time synchronisation with the data interchange:* One of the improvements consists in setting the same sample time for the real-time synchronisation task and the data interchange between the real-time simulator and HelFiCo software. This can avoid some sample instructions and events, and improve the simulation speed due to the fact that the simulation stops when a sample instruction is processed and therefore, improve the simulator performance.

- *Separated simulation approach:* This approach takes advantage of the multi-core processors (e.g. dual-core and quad-core processors) allowing the simultaneous execution of several processes. For that reason a separated simulation, where the differential equation system is divided in different parts, achieves a simulation speed improvement in a multi-core processor computer.

  This approach can be used because the model is dividable, so that one partial simulator is only responsible for a part of the system. Figure 13 shows a separated simulator where each partial simulator simulates just some heliostat rows of the field, in this case, there are 16 partial field simulators, each one simulates one heliostat row and therefore uses just one pair of FIFO files.

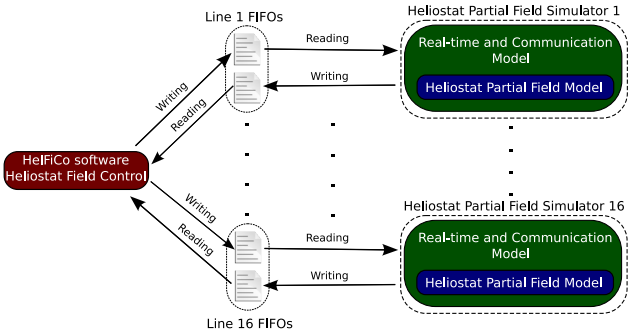

Figure 13: Separated simulator concept

Table 3 shows the CPU-Time for integration in simulation for different approaches: the traditional approach which is a complete heliostat field simulator (CS), 2 separated simulators (PS-2) and 4 separated simulators (PS-4). The 60-seconds simulation is the same for the 3 approaches and was carried

Table 3: Simulation approaches and results

Approach	CS	PS-2	PS-4
Num. Processes	1	2	4
Rows per process	16	8	4
Num. Equations per process	114,904	57,452	28,726
CPU-Time for Integration	37.40 s	21.20 s	18.22 s

out in a computer with a dual-core processor. The speed improvement from CS to PS-2 approach it was expected, because PS-2 approach can take advantage of a dual-core processor. But the PS-4 approach improves the simulation speed even when just a dual-core processor is used and this approach involves more processor context changes. Therefore this simulation speed improvement seems to be because the numerical integration in these simulation tests, do not scale properly with the number of differential equations. The numerical integrator, used in this simulations, was LSODAR [10].

5.2 HelFiCo Software

HelFiCo (Heliostat Field Control) software belongs to Aunergy ThermoSolar S.R.L, a spin-off from CIEMAT (`www.aunergy.com`). This software is in charge of manipulating and controlling each heliostat in the field according to an automatic control strategy. The main features of this software are the following ones:

- It is a generic software allowing different kind of heliostats and even different central receiver solar thermal power plants.
- The software has two different subsystems, an intuitive graphical user interface, see Figure 14, and the heliostat field control system.
- It is a scalable distributed software. This feature allows that the software can be executed in several computers to achieve scalability and robustness independently of the number of heliostats in the field.
- The software allows different communication methods with the heliostat field.
- The software can connect to both the real plant and the simulated plant, and this process is transparent for the HelFiCo software user.
- It is a multi-platform software. The Adaptive Communication Environment (*ACE*) [11] as development framework and *Qt* [12] as user interface framework have been used for the development.

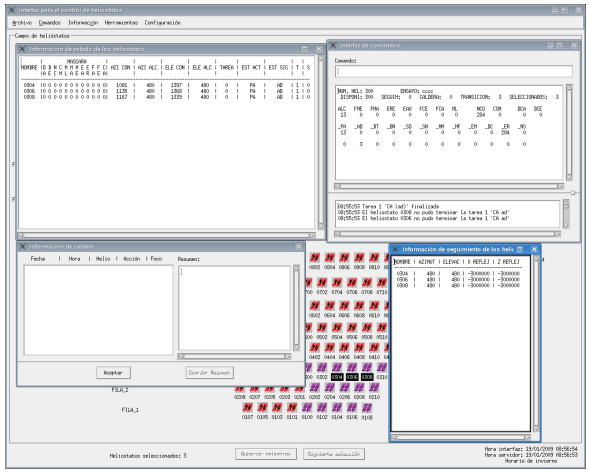

Figure 14: HelFiCo graphical user interface

6 Conclusions and Future Work

This paper explains a heliostat field real-time simulator of a central receiver solar thermal power plant (CESA-I from CIEMAT-PSA) developed using Modelica language as the modelling tool. The main goal of the developed model is providing a virtual system, with the same response as the real one, to test the control system software and to reduce the costly real experiments over the real plant. This paper also explains some techniques to improve simulation efficiency.

Future works will include a detailed explanation of dynamic heliostat models. Moreover, the incident solar radiation reflections onto the receiver to obtain the temperature distribution, will be implemented. The combination of the movement dynamic, temperature distribution on the receiver, communication model and real-time simulation will provide an efficient and practical tool to test advanced control systems for aim-point tracking to optimize the overall performance of central receiver solar thermal power plants.

7 Acknowledgment

The authors wish to acknowledge the financial support provided for this work by the CIEMAT, and the Spanish Ministry of Science and Innovation, without which this work could not have been done. The authors want to acknowledge to J. González for his help and contribution with HelFiCo software. The authors also gratefully acknowledge the comments of the reviewers.

References

[1] Bonilla, J., Roca, L., González, J., Yebra, L.J: Modelling and real-time simulation of heliostat fields in Central Receiver Plants. In: 6th Vienna International Conference on Mathematical Modelling, 2009, 2576–2579.

[2] Dynamsim, A.B.: Dymola - Dynamic Modelling Laboratory. Ideon Science Park, SE-223 70 Lund, Sweden. Online: `http://www.dynasim.se`, 2009.

[3] Frey, G.: JPAARealTime V0.1. - An open source real time option for Dymola. Department of Mechatronics Engineering, Saarland University, SaarbrÃ¼cken, Saarland, Germany. Online: `http://www.aut.uni-saarland.de/software/software.html`, 2007.

[4] Fritzon, P.: Principles of Object-Oriented Modelling and Simulation with Modelica 2.1. John Wiley & Sons, IEEE Press, 2004.

[5] García-Martín, F.J., Berenguel, M., Valverde, A., Camacho, E.F.: Heuristic Knowledge-based Heliostat Field Control for the Optimization of the Temperature Distribution in a Volumetric Receiver, In: Solar Energy, vol 66, 1999, 355–369.

[6] Garcia, P., Ferriere, A., Bezian, J.: Codes for solar flux calculation dedicated to central receiver system applications: A comparative review, In: Solar Energy, vol 82, 2008, 189–197.

[7] González, J., Yebra, L. J., Berenguel, M., Valverde, A., Romero, M.: Real-time Distributed control system for heliostat fields (in Spanish). In: XXV Jornadas de Automática, 2004.

[8] Modelica and the Modelica Association. : Modelica Standard Library, Version 2.2.1, 2007. Online: `http://www.modelica.org/libraries/Modelica/releases/2.2.1/`.

[9] Otter, M., Årzén, K-E., Dressler, I.: StateGraph – A Modelica Library of Hierarchichal State Machines. In: Proc. 4th International Modelica Conference, 2005.

[10] Petzold, L. R., Hindmarsh, A. C.: *LSODAR*. Computing and Mathematics Research Division, 1-316 Lawrence Livermore National Laboratory, Livermore CA 94550.

[11] Schmidt, D. C.: The Adaptive Communication Environment (ACE), DOC software. Washington University, University of California, Irvine, and Vanderbilt University. Online: `http://www.cs.wustl.edu/~schmidt/ACE.html`, 2007.

[12] Trolltech:A cross-platform application and UI framework. Online: `http://trolltech.com`, 2008.

[13] Yebra, L.J., Berenguel, M. and Dormido, S. and Romero, M.: Modelling and Simulation of Central Receiver Solar Thermal Power Plants. In: Proc. 2005 European Control Conference Decision and Control CDC-ECC'05. 44th IEEE Conference, 2005, 7410–7415.

