Intelligent Hybrid Control Model for Lighting Systems Using Constraint-Based Optimisation

Alie El-Din Mady, Menouer Boubekeur, Gregory Provan,
Conor Ryan, and Kenneth N. Brown

Abstract. Lighting systems consume a considerable proportion of total energy budgets, particularly for retail and public-office applications, and hence their optimisation can save considerable amounts of energy. This paper proposes an intelligent control strategy to operate the office luminance in order to enhance user comfort and reduce energy consumption. The strategy is applied to an open office scenario, where the controller and the environments are modelled using a hybrid/multi-agent platform. The developed controller uses a constraint-based optimisation technique to compute the optimal settings. We describe the different modelling steps, including the optimisation technique, and outline the simulation results and potential energy benefits of the proposed controller.

Keywords: Hybrid System, Multi-agent System, Building Automation, Constraint-Based Optimisation.

1 Introduction

An intelligent building incorporates a Building Management System (BMS) which aims to optimise energy usage while trying to optimise user comfort. One major energy consumer in buildings is lighting, which can account for up to 30% of total energy waste in some retail and public offices [1]. This energy inefficiency is due to a lack of energy-efficient lighting controllers.

Alie El-Din Mady · Menouer Boubekeur · Gregory Provan
Cork Complex Systems Lab (CCSL), Computer Science Department,
University College Cork (UCC), Cork, Ireland
e-mail: {mae1,m.boubekeur,g.provan}@cs.ucc.ie

Conor Ryan · Kenneth N. Brown
Cork Constraint Computation Centre (4C), Computer Science Department,
University College Cork (UCC), Cork, Ireland
e-mail: {c.ryan,k.brown}@4c.ucc.ie

E. Corchado et al. (Eds.): SOCO 2010, AISC 73, pp. 249–259.
springerlink.com

The aim of our research is to define a methodology for efficient modelling and integration of building management system services, in particular lighting and Heating, Ventilating, and Air Conditioning (HVAC) systems. We assume that building automation models can be represented using hybrid systems models [2], since hybrid systems can represent both the discrete-valued and continuous differential-equation-based relations essential for such models.

In this paper we show how we can use component-based hybrid systems to model and simulate an intelligent lighting controller. The developed lighting system tracks the presence of people in the different controlled areas and allows users to express preferences for interior lighting levels. The control system accommodates such preferences for all occupants within a zone, by optimising a global preference/energy function using a constraint-based optimiser to compute the optimal light luminance levels specified by the user preferences, and the power consumed by the artificial light and the blinding actuators. The centralized controller then maintains this setpoint by adjusting window blinds to control the exterior light levels, and by dimming the interior lights.

The remainder of the paper is organized as follows: Section 2 introduces our modelling platform and describes the system scenario specification, the control strategy and the corresponding hybrid modelling for the control and its environments. The constraint-base optimisation technique is explained in Section 3. Section 4 outlines the simulation results. Finally, we conclude the paper in Section 5.

2 Hybrid Modelling for Intelligent Lighting System

We now apply our design methodology to develop an intelligent controller for a lighting system. In this section we introduce our modeling framework, describe the scenario specification for the lighting system model, the control strategy and finally the corresponding hybrid modelling.

2.1 Hybrid Platform for Building Control

Building systems are a perfect example of hybrid systems, where continuous and discrete dynamics must be used for modelling. For example, heat dissipation and luminosity follow a continuous dynamics whereas presence detection is a discrete nature. In our work we show how we can use a component-based hybrid-systems modelling framework to generate models for simulation and verification.

To implement the hybrid systems for building models, we using the CHARON tool [3]. We assume that we can create/redesign a system-level model by composing components from a component library [[4], [5]]. We call a well-defined model fragment a component. We assume that each component can operate in a set of behaviour-modes, where a mode *M* denotes the state in which the component is operating. For example, a pump component can take on modes nominal, high-output, blocked and activating.

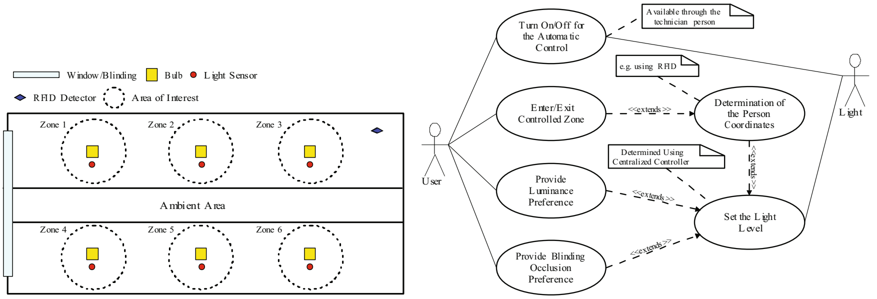
Our ongoing research work consists of developing an integrated platform for intelligent control of building automation systems. This platform provides, among other features, predictability, reconfiguration, distribution and building energy optimisation.

The system design flow starts by defining relevant scenarios to be operated within the building. These scenarios are defined using the Unified Modelling Language (UML) [6]. The UML models are interpreted using specific models for simulations and analysis purposes. At this level we allow an optimisation loop to optimise the model at an early stage of the development. When the simulation gives satisfactory results, the models are auto-translated into embeddable code to be deployed over a distributed sensor/actuator network [7]. The integration process is performed through the implementation of a model-/service-based middleware [8] platform allowing components connection and data exchange. All the different components of the architecture collaborate with the requirements module.

The main features of our platform will be illustrated through an example of a lighting system for an office area. This example illustrates the combination of discrete-event behaviour (presence detection, light actuation levels) and hybrid properties for the luminosity control, i.e., where both discrete and continuous aspects are considered.

2.2 Scenario Specification

We have adopted the architecture shown in Fig. 1(a) for our work. We focus on an open office area, which contains 6 controlled zones, where each zone contains one artificial light and one light sensor. One Radio-Frequency Identification (RFID) receiver is used to cover the whole area; there is one window/binding in the left boarder of the conceded area and a fix number of predefined person positions. For the lighting model we integrate blinding and lighting controls. In order to enhance the efficiency of the resulting control model, an optimisation technique has been implemented, as explained in Section 3.


Window/Blinding
Bulb
Light Sensor
RFID Detector
Area of Interest
Zone 1
Zone 2
Zone 3
Ambient Area
Zone 4
Zone 5
Zone 6
Turn On/Off for the Automatic Control
Available through the technician person
e.g. using RFID
Enter/Exit Controlled Zone
<<extends>>
Determination of the Person Coordinates
Light
User
Determined Using Centralized Controller
<<extends>>
Provide Luminance Preference
<<extends>>
Set the Light Level
Provide Blinding Occlusion Preference
<<extends>>

(a) Model Specification (b) UML Use-case

Fig. 1 Lighting System Specification

As a summary, the lighting control scenario, as described in Fig. 1(b), behaves as follows:

1. The user can switch on/off the automatic lighting system for several zones, or for all the system through a technician.
2. The users provide their light luminance preferences.
3. A person is tracked in each zone using RFID, and his preferences are considered wherever he is located.
4. An optimisation engine receives the user preferences and sends back the optimal settings.
5. The controller controls the artificial light and the blinding actuators in order to reach the user preferences considering the daylight luminance and the light interferences coming from the adjacent zones.

2.3 *Control Strategy Modelling*

Fig. 2 shows the agents of the control model and its interactions with the environment agents. The controller follows the following scenario in order to control the light intensity:

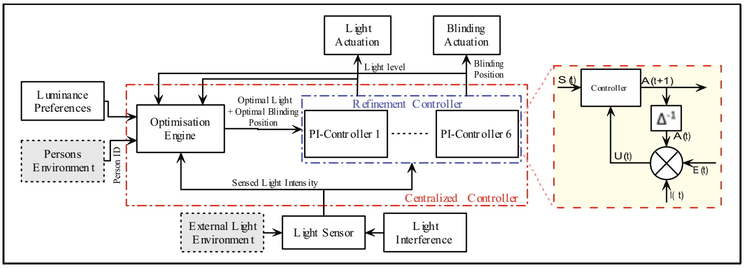


Fig. 2 Control Model

1. The optimisation engine receives the user preferences for each person and its position, sends the optimal light luminance and blinding position back to the refinement controller in order to refine the actuation values using a PI-Controller.
2. The refinement controller actuates the artificial light and the blinding position accordingly, then goes to 1 if the preference has been changed or a significant change in the sensed light occurs otherwise the PI-Controller actuates relying on the external light and the light interference changing.

The PI-Controller is used to predict the next actuation setting for the lighting dimming level in a close loop fashion [9] using Eq. 1. The PI-Controller has two main status, first is unstable when the difference between the sensed light intensity and the optimal one is greater than 70 Lux (one light actuation level), and secondly, is stable, if the difference is less than or equal to 70 Lux.

$$A(t+1) = A(t) + \alpha \tag{1}$$
$$U(t) = A(t) + E(t) + I(t)$$

$$\alpha = \begin{cases} \lfloor \frac{U(t)-S(t)}{\varepsilon} \rceil \times -\varepsilon, & \forall\, U(t)-S(t) > \varepsilon \\ \lfloor \frac{S(t)-U(t)}{\varepsilon} \rceil \times \varepsilon, & \forall\, S(t)-U(t) > \varepsilon \\ 0, & \forall\, |S(t)-U(t)| \leq \varepsilon \end{cases}$$

Where, $A(t)$ is the actuation setting for light/blinding actuators, $E(t)$ is the daylight intensity (Lux), $I(t)$ is the interference light intensity (Lux), $U(t)$ is the sensed light intensity (Lux), $S(t)$ is the optimal preference settings and ε is the luminance level produced from a single dimming level (70 Lux).

2.4 Hybrid Modelling

In the Charon modelling shown in Fig. 2, two types of agent have been used to model the control- and environment agents.

Control Agents : One main agent is used for the refinement controller, such that one subagent is used to refine the actuation values in each zone using a PI-Controller as depicted in Fig. 3(c). Another agent is used to call the optimisation engine; it follows the behavioral mode depicted in Fig. 3(b). This agent is triggered whenever the user preferences change or a significant change in the sensed light is observed. Finally, the sensor agent has been modelled as shown in Fig.3(a), as it updates the internal light value every sampling period, based on the actuation value, the light interference and the daylight light coming to the sensor. It considers an intensity attenuation factor of $1/r^2$, where r is the distance from the light source to the sensor.

Three environment agents have been used to verify the control behaviour as following:

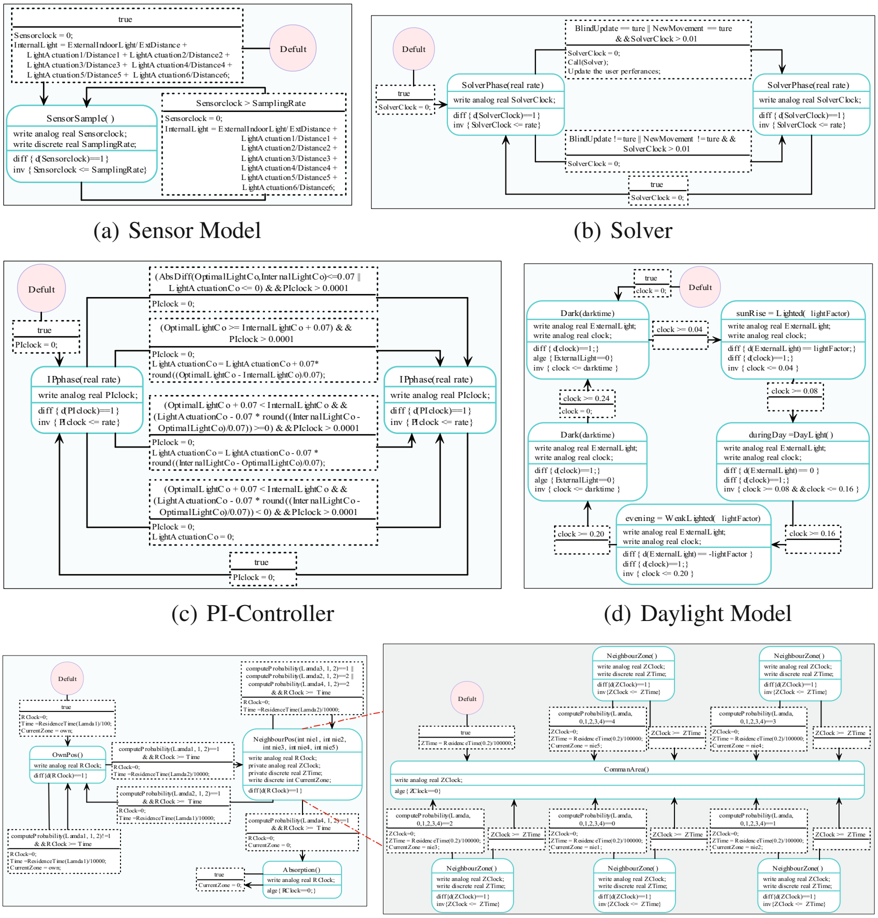
Person Movements : One agent is used to model a person's movements. This agent uses a Markov chain to model the person presence in the zones using a Phase-type Distribution [10]. As shown in Fig. 4, λ is the transition probability between each zone and the ambient area. The time consumed (t) in each zone follows Eq. 2. Fig.3(e) shows the hierarchal hybrid automata for the Markov chain model. When a person moves from his zone (current zone) to the next zone, the model goes to the deeper level in order to reflect his transition to the other zones. If the user moves to the absorption state, that means he goes out of the controlled area.

$$f(t,\lambda) = 1 - e^{-\lambda t} \tag{2}$$

Daylight Intensity : In the daylight model shown in Fig. 3(d), five periods have been modelled as a first order differential equation with a constant slope using linear hybrid automata [11]. During the first and last four hours of the day, the daylight slope and luminance are equal to zero, while during the second four hours the slope is equal to 100 which means that the maximum intensity in the day is 4000 Lux. In the next eight hours the slope is equal to zero and then goes to -100 in the following four hours in order to reach zero luminance again at the end of the day. The light intensity that comes to the controlled zone is a percentage of the daylight intensity; this percentage relies on the dimensions of the window. In this model, 8% of the daylight is considered as the external light coming into the controlled zone [9].

Window Blinding Occlusion : One Charon agent is used to model the blinding occlusion as an algebraic equation. It calculates the daylight percentage coming to the controlled zone as a linear proportion from the blinding position as following

```
alge{ExternalIndoorLight ==
        ExternalLight*(BlindAct/TotalControlLevels)}
```


true
Sensorclock = 0;
InternalLight = ExternalIndoorLight/ExtDistance + LightActuation1/Distance1 + LightActuation2/Distance2 + LightActuation3/Distance3 + LightActuation4/Distance4 + LightActuation5/Distance5 + LightActuation6/Distance6;
Defult
SensorSample( )
write analog real Sensorclock;
write discrete real SamplingRate;
diff { d(Sensorclock)==1 }
inv { Sensorclock <= SamplingRate}
Sensorclock > SamplingRate
Defult
true
SolverClock = 0;
SolverPhase(real rate)
write analog real SolverClock;
diff { d(SolverClock)==1 }
inv { SolverClock <= rate}
BlindUpdate == ture || NewMovement == ture && SolverClock > 0.01
SolverClock = 0;
Call(Solver);
Update the user perferances;
BlindUpdate != ture || NewMovement != ture && SolverClock > 0.01
Defult
PIclock = 0;
(AbsDiff(OptimalLightCo,InternalLightCo)<=0.07 || LightActuationCo <= 0) && PIclock > 0.0001
IPphase(real rate)
write analog real PIclock;
diff { d(PIclock)==1 }
inv { PIclock <= rate}
(OptimalLightCo >= InternalLightCo + 0.07) && PIclock > 0.0001
LightActuationCo = LightActuationCo + 0.07* round((OptimalLightCo - InternalLightCo)/0.07);
(OptimalLightCo + 0.07 < InternalLightCo && (LightActuationCo - 0.07 * round((InternalLightCo - OptimalLightCo)/0.07)) >=0) && PIclock > 0.0001
LightActuationCo = LightActuationCo - 0.07 * round((InternalLightCo - OptimalLightCo)/0.07);
(OptimalLightCo + 0.07 < InternalLightCo && (LightActuationCo - 0.07 * round((InternalLightCo - OptimalLightCo)/0.07)) < 0) && PIclock > 0.0001
LightActuationCo = 0;
clock = 0;
Defult
Dark(darktime)
write analog real ExternalLight;
write analog real clock;
diff { d(clock)==1; }
alge { ExternalLight==0}
inv { clock <= darktime }
clock >= 0.04
sunRise = Lighted(lightFactor)
diff { d(ExternalLight)== lightFactor; }
inv { clock <= 0.04 }
clock >= 0.08
duringDay =DayLight( )
diff { d(ExternalLight)== 0 }
inv { clock >= 0.08 && clock <= 0.16 }
clock >= 0.16
evening = WeakLighted(lightFactor)
diff { d(ExternalLight)== -lightFactor }
inv { clock <= 0.20 }
clock >= 0.20
clock >= 0.24
OwnPos( )
write analog real RClock;
diff{d(RClock)==1}
NeighbourPos(int nie1, int nie2, int nie3, int nie4, int nie5)
private analog real ZClock;
private discrete real ZTime;
write discrete int CurrentZone;
computeProbability(Lamda1, 1, 2)==1 && RClock >= Time
Absorption()
alge{ RClock==0;}
CurrentZone = 0;
NeighbourZone( )
write analog real ZClock;
write discrete real ZTime;
diff{d(ZClock)==1}
inv{ZClock <= ZTime}
ZClock >= ZTime
CommanArea()
alge{ ZClock==0}
ZTime = ResidenceTime(0.2)/100000;

(a) Sensor Model

(b) Solver

(c) PI-Controller

(d) Daylight Model

(e) Person Movements

Fig. 3 Linear Hybrid Automata Models

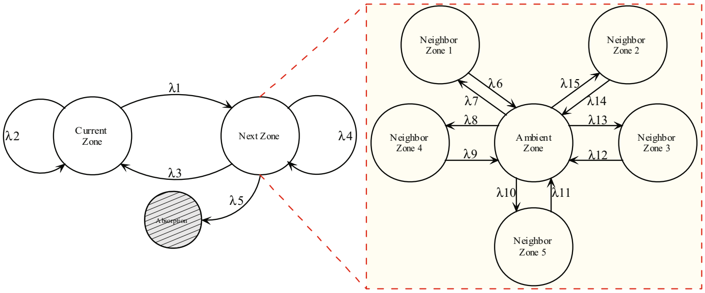


Fig. 4 Markov Chain for Person Movements

3 Constraint-Based Optimisation

The purpose of the optimisation engine is to compute settings for the lights which optimise both the energy use and the occupant satisfaction levels. The computation is based on the inferred external sunlight and the stated occupant preferences, and uses an idealised model of the domain. The derived settings together with the preferences are sent to the controller as initial actuator settings. The controller then controls the lighting around these initial values. When the occupants change, or the actuated levels diverge significantly from the initial values, the problem is re-specified to the optimisation engine, which re-computes and issues new settings.

We model the building environment, the actuated lights and their effect on the environment as a Constraint Optimisation Problem. A constraint problem [12] consists of a set of variables, a domain of possible values for each variable, a set of constraints over the assignment of values to variable which restrict the values that may be assigned simultaneously, and an objective function over the assignments. A solution is an assignment of one value to each variable, such that no constraint is violated. An optimal solution is one with the highest objective value. Solutions may be obtained by any suitable method, including backtracking and logical reasoning, mathematical programming, or local search.

The variables in our model include the setpoints of the actuated lights, the blinding level, the lux levels at the zone sensors, and the lux level of the external light. The domains for the light setpoints are the 11 integer values from 0 (off) to 10 (full power) and the domain for the blinding is the 5 integer values from 0 (fully closed) to 4 (fully open). The constraints relate the actuation values to the lux levels (and so in our model the setpoints are decision variables, the external light is a constant, and the sensed lux levels are dependent variables). We assume that each light source contributes a fraction of its luminance to each zone, using the same underlying model in the simulator. A solution is then a set of actuation points and lux levels such that the constraints are satisfied. When the optimisation model is activated, the controller passes it the actuation values and the sensed lux levels for the current state; from this, we can infer the current value of external light.

We associate with each individual occupant a preference curve, which associates a degree of satisfaction from 0 to 1 with each possible lux level. These preference curves may be explicitly stated by the occupant, or learned from their actions. We then create a preference curve for each zone, by averaging the preference curves of the occupants in each zone. From this zone preference curve we can determine the overall preferred lux level in each zone. For a candidate solution (i.e. tuple of actuation points), we can extract the value of the sensed lux level for each occupied zone, and compute the deviation between the preferred and sensed levels. In addition, for each actuation setting, we can compute the power required to achieve it, and we then combine these two measures as a weighted sum to get the objective value of the solution. Fig. 5 shows the complete process, including the inference of the external light. Our aim is then to search through the space of possible assignments to find the one which maximises the objective value. We do this using backtracking search interleaved with constraint propagation, using the min-dom and min-value variable and value selection heuristics. The model is implemented in and solved with CP-Inside [13]. We find the optimal in, on average, 250 milliseconds.

Note that the optimisation model is not an accurate control model. It is a simplified model of actuators and of their effect on the environment, and it assumes the actuation is perfect, the propagation of light is uniform, and that the sensors are perfect. It also does not account for small variations in the external light. Instead, from the optimal solution we derive the intended lux levels for each zone, and pass them to the controller as the target lux levels. The values for the lights are passed as initial settings, and the controller then controls the actuated lights in order to achieve the target lux levels. The controller thus handles the inaccuracies in the actuation and sensing, and any small variations in external light. When the controller changes the set points of the lights to such an extent that a light is 2 or more set points away from its initial setting, we assume that the external light has changed, and so we re-state the problem to the optimiser, which computes a new set of intended lux levels and a new initial settings for the lights. Similarly, when we detect that an occupant has left, arrived, or changed position, we again pass this new problem to the optimiser.

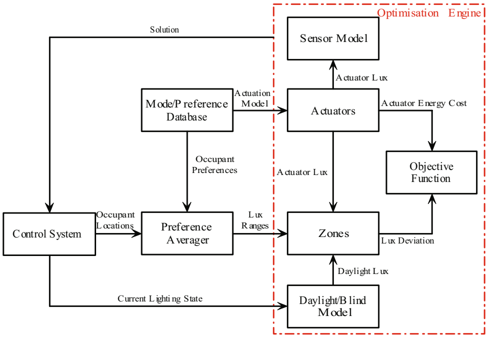


Fig. 5 Opimisation Process

4 Simulation Results

The Charon model described earlier and its environment have been simulated using the Charon simulation tool-set. In this section we provide the simulation results and evaluate the power saving comparing to a typical control technique used in building automation.

4.1 Lighting Control

In order to observe the control- and the environment behaviour, Fig. 6(e) shows the external light coming to the controlled area and how it is affected by the blinding occlusion changes of Fig. 6(f) (5:06 - 5:24pm and 7:24-7:54 pm) and the clouds move (11:36am-4:54 pm). However, Fig. 6(c) shows the person movements among the zones and their effect on the optimal light that calculated by the optimisation engine for each zone, as depicted in Fig. 6(b).

To evaluate the control strategy, we consider 2 zones: the first one (Zone 1) is highly affected by the daylight whereas the other one (Zone 6) is less affected. Regarding internal light in Zone 1 (shown in Fig .6(a)), when there is no presence in zone 1, the internal light is affected only by the external light and the interferences from the neighbour zones. However when a person comes inside, internal light 1 gets actuated to reach the optimal light decided by the optimiser. Zone 6 follows the same routine, but with reduced daylight influence; for example at 9:42 pm, when zone 6 is unoccupied, internal light 6 equals to 82 Lux which mainly comes from the light interference.

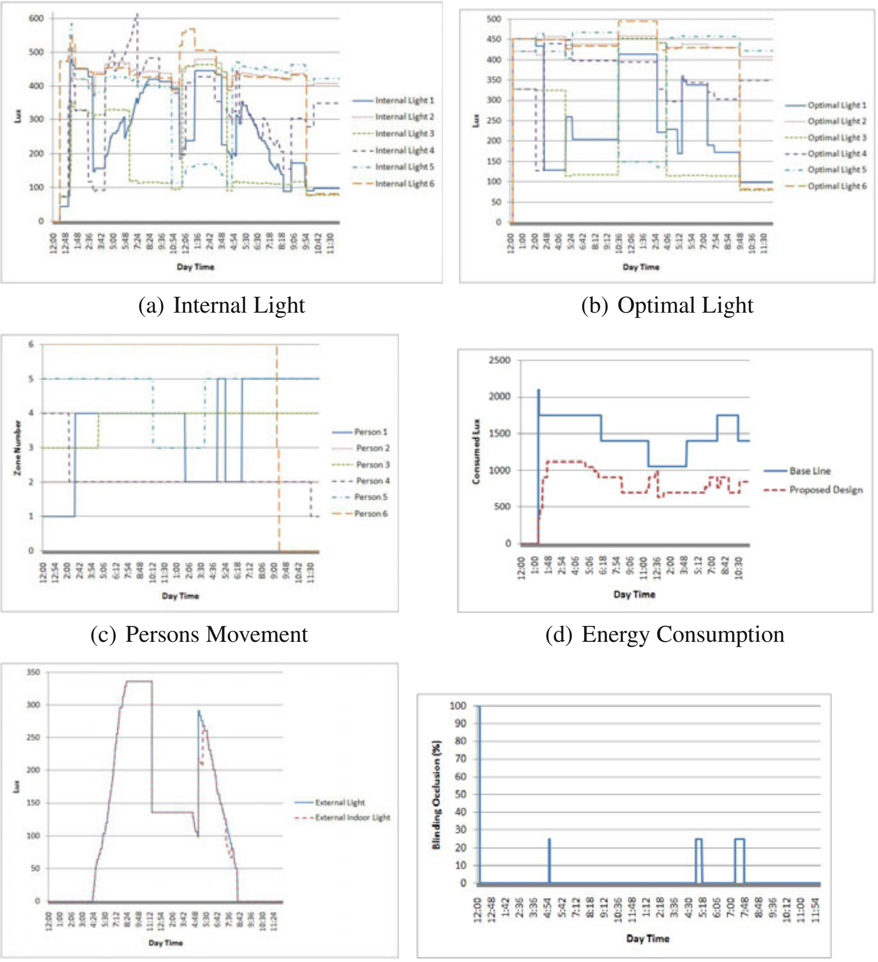
4.2 Energy Saving

In order to evaluate the potential energy efficiency of the proposed control strategy, we have considered a typical control strategy used in building automation as base-line for comparison. The base-line model uses Passive Infrared (PIR) sensor for presence detection in order to switch on the artificial light to a predefined luminance level. In our case, we consider 350 Lux as the optimal light in the entire zone since it is almost the average optimal intensity requested as in Fig. 6(b). Due to the fact that energy consumption has a linear relation with the consumed Lux over time, we have compared the consumed Lux in each case as shown in Fig. 6(d). The results show that the proposed control strategy reduces energy consumption by 42% in comparison to the baseline model.

5 Conclusion

In this article we have proposed a hybrid/muti-agent model for an intelligent automated lighting system. The control strategy maintains user comfort through systematically tracking occupants in each zone in order to integrate their preferences. The control system incorporates a constraint-based optimiser that computes the optimal setting, thereby optimising energy usage while providing adequate user comfort. The simulation results shows that the proposed controller saves around 42% of the energy consumption compared to a standard baseline control strategy.

As future work, we intend to deploy a demonstration of the developed system in the Environmental Research Institute (ERI) building, which is the ITOBO Living Laboratory [14].



(a) Internal Light

(b) Optimal Light

(c) Persons Movement

(d) Energy Consumption

(e) External Light

(f) Blinding Actuation

Fig. 6 Model Simulation Results

Acknowledgements. This work was funded by Science Foundation Ireland (SFI) grant 06-SRC-I1091.

References

1. The Interlaboratory Working Group on Energy-Efficient and Clean-Energy, Scenarios for a Clean Energy Future: Interlaboratory Working Group on Energy-Efficient and Clean-Energy Technologies (2000)
2. Labinaz, G., Bayoumi, M.M., Rudie, K.: Modeling and Control of Hybrid Systems: A Survey. In: Proc. of the 13^{th} Triennal World Congress, San Francisco, USA (1996)

3. Charon: Modular Specification of Hybrid Systems, http://rtg.cis.upenn.edu/mobies/charon/
4. Gössler, G., Sifakis, J.: Composition for Component-Based Modeling. In: de Boer, F.S., Bonsangue, M.M., Graf, S., de Roever, W.-P. (eds.) FMCO 2002. LNCS, vol. 2852, pp. 443–466. Springer, Heidelberg (2003)
5. Keppens, J., Shen, Q.: On compositional modeling. In: The Knowledge Engineering Review, pp. 157–200 (2001)
6. Booch, G., Rumbaugh, J., Jacobson, I.: The Unified Modeling Language User Guide. Addison Wesley, Reading (1998)
7. Mady, A., Boubekeur, M., Provan, G.: Compositional Model-Driven Design of Embedded Code for Energy-Efficient Buildings. In: 7^{th} IEEE International Conference on Industrial Informatics (INDIN 2009), Cardiff, UK (2009)
8. Romer, K., Kasten, O., Mattern, F.: Middleware Challenges for Wireless Sensor Networks. In: ACM SIGMOBILE Mobile Computing and Communications Review (2002)
9. Kolokotsa, D., Pouliezos, A., Stavrakakis, G., Lazos, C.: Predictive control techniques for energy and indoor environmental quality management in buildings. In: Building and Environment (2008)
10. Marshall, A.H., McClean, S.I.: Using Coxian Phase-Type Distributions to Identify Patient Characteristics for Duration of Stay in Hospital. In: Health Care Management Science (2004)
11. Henzinger, T.: The theory of hybrid automata. In: Proc. 11^{th} Annual IEEE Symposium on Logic in Computer Science (LICS 1996), pp. 278–292 (1996)
12. Dechter, R.: Constraint Processing. Morgan Kaufmann Publishers, San Francisco (2003)
13. Feldman, J., Freuder, E., Little, J.: CP-INSIDE: Embedding Constraint-Based Decision Engines in Business Applications. In: Integration of AI and OR Techniques in Constraint Programming for Combinatorial Optimization Problems (2009)
14. Environmental Research Institute, http://www.ucc.ie/en/ERI/