Final Degree Project

Degree in Industrial Technology Engineering

CENTRALIZED AND NON-CENTRALIZED MODEL PREDICTIVE CONTROL OF A MULTIZONE BUILDING

REPORT

Author: Pablo Delgado Alonso
Director: Vicenç Puig Cayuela

Date: June 2017

Escola Tècnica Superior
d'Enginyeria Industrial de Barcelona

Abstract

Nowadays, reduction in energy use has become an important issue by the vast majority of institutions in all the world. Although the energy efficiency of systems and components for heating, ventilating, and air conditioning (HVAC) has improved considerably over recent years, there is still potential for substantial improvements. Consequently, this project deals with an advanced control technique, model predictive control, that can provide significant energy savings in comparison with conventional, non-predictive techniques.

Nevertheless, the main goal is to try a comparison between two possible approaches to obtain such building energy control problem: (1) Centralized control, and (2) Distributed control. To accomplish this, mathematical software MATLAB has been used. Also YALMIP, which is a free MATLAB toolbox for rapid prototyping of optimization problem, has been used.

Keywords: Energy savings; Building control optimization; Thermal comfort, Centralized control, Distributed control, Distributed architectures, Model Predictive Control.

SUMMARY

1. GLOSSARY

1.1. Abbreviations

- **MPC**: Model Predictive Control
- **EMPC**: Economic Model Predictive Control
- **DMPC** : Distributed Model Predictive Control
- **HVAC**: Heating Ventilation Air-Conditioning
- **VAV**: Variable Air Volume
- **FCU**: Fan Coil Unit
- **AHU**: Air Handle Unit
- **LSS :** Large Scale System

2. PREFACE

2.1. Origin of the project

The origin of this project starts with a first meeting that has held in February 2017 with my tutor Vicenç Puig. He was working on an innovative project related with energy optimization with cooperation with an another faculty in Nancy, France.

The proposed idea of analysing and learning more about a new control field seemed very interesting to me. I was curious to learn more about the automatic control, after having studied the subject of automatic control of my degree. More specifically, I was really interest on learning the internal structure of the controllers and how they work.

2.2. Academic requeriments

To carry out this project, it was necessary to learn more about the concept of predictive control, state space and decomposition in Nonlinear programming. A previous and in-depth study was required of how the control strategy works, the energy optimization problem that must be formulated in each case, the use of mathematical models within the controllers and how to deal with the non-linearity of the predictive models.

On the other hand, the knowledge acquired in subjects such as Optimization and Simulation and Automatic Control have been very useful for carrying out this work. Finally, the learning of Matlab software, Yalmip and other solvers such as Gorubi has been indispensable.

2.3. Motivation

The motivation to start that project was clear: this work is within an innovative field. There are several interesting topics of relative relevance and that in the future may have a very positive impact for our society. I also consider that doing a job related to the automatic control is very interesting since it has many applications in this case environmental, saving energy consumption, sustainable consumption ...

3. INTRODUCTION

Energy savings in buildings have gained a lot of attention in recent years. Most of the research is focused on the building construction or alternative energy sources in order to minimize primary energy consumption of buildings. By contrast, this project deals with an advanced process control technique called model predictive control (MPC) that can take advantage of the knowledge of a building model and estimations of future disturbances to operate the building in a more energy efficient way.

3.1. Objectives of the project

Two non-centralized methods building energy optimization are presented in order to make a comparison with a centralized solution.

Moreover, other objectives of this project are:

i) Evaluate model predictive control energy savings potential on building simulation studies.

ii) Develop a large-scale centralized formulation that takes into account thermal comfort of occupants and weather forecast information.

iii) Develop and evaluate alternative architectures that takes into account techniques to solve modular subproblems which come from the centralized solution.

Finally, this project was didactic purpose because it wants to introduce new control techniques that are currently emerging in the automatic control sector.

3.2. Scope of the project

First, the concept of model predictive control will be introduced and the main characteristics of this advanced control technique. Then, a discussion will be introduced of how a considerable building system can be treated. Secondly, two important sections will explain not only the centralized approach given to a heating and ventilation problem in the building but also two innovative non-centralized techniques: Optimal Condition Decomposition and Sensitivity-Based Coordination.

4. MODEL PREDICTIVE CONTROL

4.1 Introduction

Model based Predictive Control (MPC) is a control technique for dynamic systems that computes optimal control set points in order to minimize a predefined cost. For this, the controller contains a model, based on dynamical system and its predicted future evolution, that is used in an optimization routine.

The control objective and the mathematical model is formulated as a real-time optimization problem that repeatedly generates control inputs. The objective may be related to minimizing operational costs, maximizing profit or forcing the system to follow a pre-computed set point trajectory. Only the computed inputs associated with the current time step is actuated on the physical system. A new current model state is estimated regularly when new measurements are available and the real-time optimization procedure is repeated. It is important to realize that this repeated optimization procedure provides closed-loop feedback and enables the MPC to counteract model uncertainties and external disturbances giving a certain robustness to the system. This principle is illustrated in Figure 4.1 and is also often referred to as Receding Horizon Control (RHC), explained in detail in next section [1].

Traditionally MPC was designed using an objective function that penalizes deviations of a given set point. But now with the economic control MPC has emerged as a general approach with multiple implementations and numerical stability properties. It has increased its prominence in recent years for climate control systems in buildings.

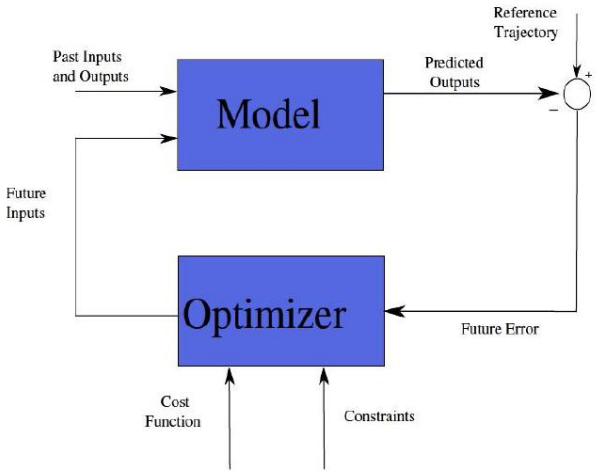


Fig. 4.1 Schematic of a typical close-loop basic MPC structure

4.2 Principles of MPC

Model Predictive Control is a multivariable control algorithm that is based on:

- **Explicit use of an internal dynamic process model**

One of the crucial contributors to the quality of the control is a well identified model which will be later on used for control in MPC algorithm. It describes the expected behaviour of the system and it can be formulated as lineal or non-lineal, continuous time or discrete time and in state variables or input-output approach.

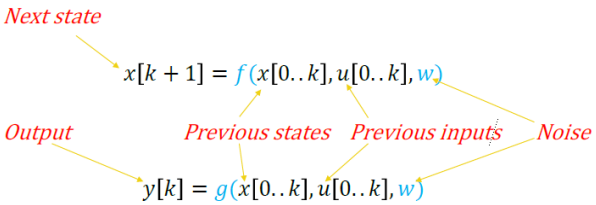


Fig. 4.2 Discrete-time and state space form system formulation

The application of models pursues the following key points:

- Prediction of future process output behaviour.
- Determination of the best future input manipulations to drive the process to optimum conditions
- Feedforward compensation of disturbances
- Respecting operating constraints and determination of optimum conditions
- Handling of non-linearities

Moreover, there are several completely different approaches to system identification:

- Physical modelling using equations.

  Also called white-box concept, it is the most common building engineers approach. It is based on using material properties, textbooks and blueprints or by using specific software. It requires availability and processing of a large amount of building-specific information derived directly from detailed geometry and construction data.

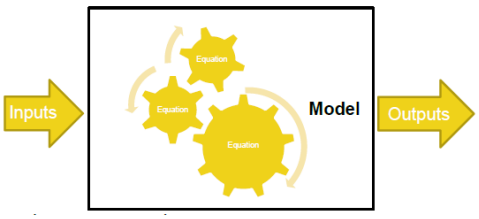


Fig. 4.3 White-box system identification approach.

- Statistical modelling using data.

The model structure and parameters are identified in a statistical-empirical manner from on-site measurements. This modelling strategy is called black-box approach and it is conceptually simple, but technically tricky, and it crucially depends on the availability of appropriate input data sets that encompass sufficient long sequences of all relevant excitation-response signal pairs. These are very hard to obtain from a real building during normal operation.

Fig. 4.4 Black-box system identification approach.

- Hybrid using both data and equations.

This approach, known as semi-physical or grey-box, describes a building's thermal dynamics based on a thermal resistance capacitance (RC) network.

Grey-box system identification is a technique which pre-defines the model structure based on physical knowledge but which optimizes its parameter values such that the model response fits some measurement data. It presents an analogue to an electric circuitry, with temperature gradients and heat fluxes replacing electric potentials and currents. A plausible model structure (RC network topology) is first specified a priori, and then the model parameters are identified from measurements. The advantage of this approach is that basic knowledge of possible thermal interactions (e.g., neighbourship of building zones) can easily be introduced. However, the parameter identification is far from trivial.

Physics ⟹ Structure of the model

Data ⟹ Numerical values of parameters

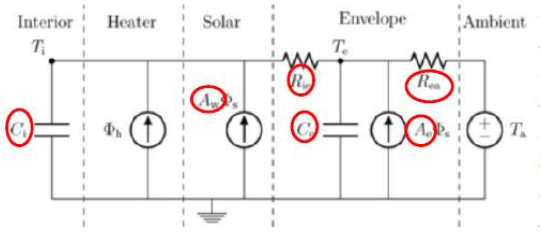


Fig. 4.5 Grey-box system identification approach: RC network topology.

➢ **Predictive evolution of the system behaviour**

The explicit use of a model and the use of historical data helps to predict future outputs of the process over a prediction horizon.

➢ **Cost function (objective function)**

It is the function that indicates the criteria to optimise. Is a define positive function that express associate cost J of system over the receding prediction horizon.

The cost function generally serves two purposes:

- Stability. It is important to choose a cost function structure that guarantee stability for the closed loop system. This requirement can be relaxed for stable systems with slow dynamics, such as buildings, which leaves the designer free to select the cost strictly on a performance basis.

- Performance target. The cost is generally, but not always, used to specify a preference for one behaviour over another, e.g., minimum energy or maximum comfort.

➢ **Receding horizon strategy with recomputation in the next time step**

The horizon for which prediction is performed moves ahead in time at each sampling instant. It means that at each iteration only the first step of the calculated control strategy is implemented and the control signal is calculated again, thus, in fact, the prediction horizon keeps being shifted forward. This strategy is known as receding horizon control (RHC).

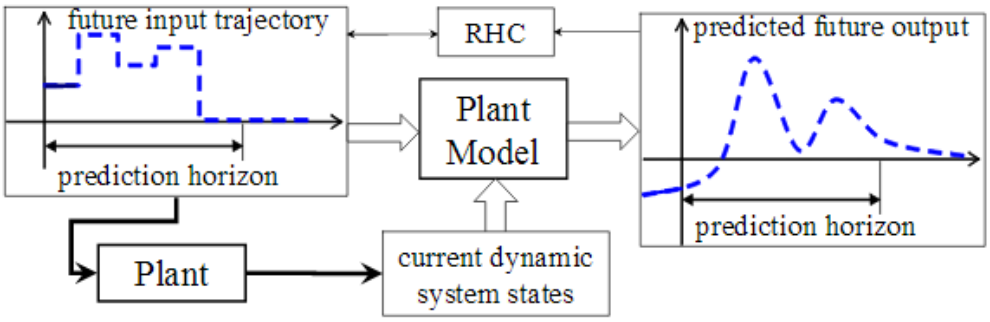


Fig. 4.6 Receding horizon strategy.

➢ **Handling of constraints**

Constraints indicate the limits where the evolution of the system must take place. The evolution of the system signals must not exceed certain restrictions that, because of physical limits or safety, they are imposed to the system.

All physical systems have constraints:

- Physical constrains (actuator limits).
- Performance constraints (settling time, overshoot).
- Safety constraints (temperature/pressure or voltage/current limits).

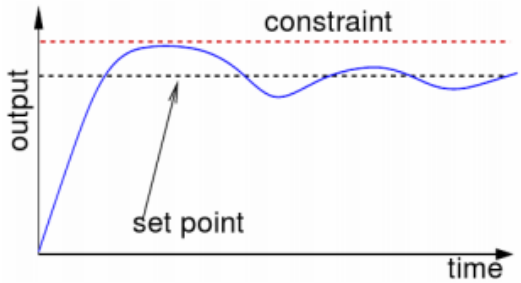


Fig. 4.7 Relations between output, set point and constraints

Current needs of working at some set points, generally for economic reasons, near to the admissible physical limits of the system has induced the incorporation of the above mentioned restrictions in the synthesis of the controllers. So, the ability to specify constraints in the MPC formulation and to have the optimization routine handle them directly is the key strength of the MPC approach. There can be constraints on the states or the output, as well as on the input. Linear constraints are the most common type of constraint, which are used to place upper and lower bounds on system variable.

$$u_{min,k} \leq u_k \leq u_{max,k}$$

4.3 MPC control strategy

The basic idea of MPC is to exploit a model of the system to simulate its future evolution over a prediction horizon and compute an optimal control action (with respect to a predefined cost function) by solving, at each decision time instant, an open-loop optimisation problem in a receding horizon fashion. The result is a trajectory of inputs and states into the future, respecting the formulate dynamics and constraints.

The MPC strategy comprises the following basic steps:

1. Calculate system output Y: Each time t the future outputs are predicted in an open-loop manner using the dynamical model provided information about past inputs, outputs and future signals, which are about to be calculated.

2. Calculate control signals U: The future control signals are calculated by optimizing the objective function, and the chosen criterion, which is usually in the form of quadratic function. The criterion constituents can be as follows:

- Errors between the predicted signal and the reference trajectory r(k)
- Control effort
- Rate of change in control signals

Predictions y (t + k | t) for k = 1 ... N depend on known values at time t (past inputs and outputs) and future control signals u (t + k | t), k = 0 ... N -1 which are intended to send to the system. The result of the optimization is an optimal sequence of future actions:
u*= [u (t | t), u (t + 1 | t), u (t + 2 | t), ..., u (t + N -1 | t)] T

Additionally, an assumption can be formulated such as the control signal may be constant at a certain moment.

3. Apply U to system: The first component of the control sequence u (t | t) is sent to the system, whilst the rest of the sequence is disposed. The signal u (t | t) is applied and rejected all other u (t + 1 | t).

4. Acquisition of new measures of the system: At the next time instant, new output y(t + 1 | t +1) is measured, first component u(k+1) is applied to the system and the rest is disposed. This principle is repeated continuously at the next sampling time by moving the estimation and regulation window, so a new current model state is estimated regularly when new measurements are available. This optimization procedure provides closed-loop feedback and enables the MPC to counteract model uncertainties and external disturbances.

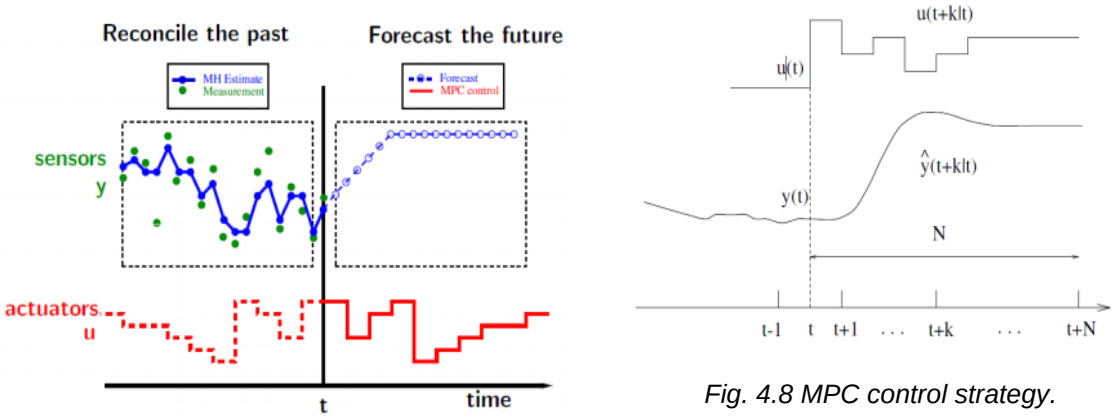


Fig. 4.8 MPC control strategy.

4.4 Prediction and control horizon

There are two important horizons in MPC, both of which are expressed in terms of sampling instants [2].

- **The prediction horizon**: It is the span of time for which the plant outputs are predicted. The size of the prediction horizon is generally limited by computation speed

- **The control horizon**: It is the number of control inputs that are calculated in the prediction computation, and is always smaller than the prediction horizon.

Therefore, it is important to choose the control horizon such that the difference between the control and prediction horizons is as least as long as it takes for all dynamics in the system to settle out

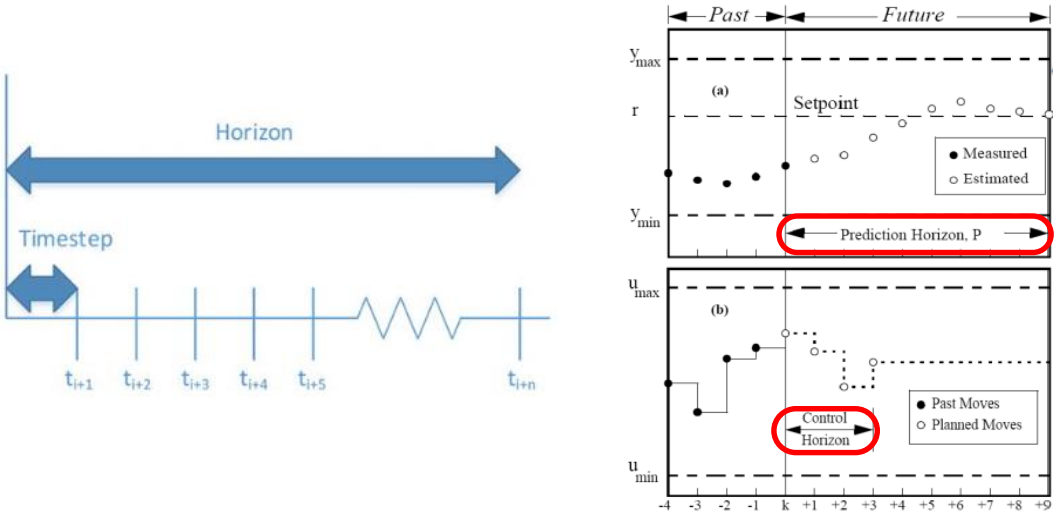


Fig. 4.9 Simulation time step and forecast prediction and control horizon.

4.5 MPC common analogies

Once it has been presented the basic principles of model predictive controllers, some easy analogies are introduced in order to illustrate clearly this powerful control strategy. These examples have common elements as car driving in a professional circuit, chess and an orientation GPS system.

- **Car driving in a professional circuit.**

Prediction model	Look forward and plan path based on: -Road conditions -Upcoming corners -Abilities of pilot, etc.
Constraints	Stay on road -Do not skid -Limited acceleration
Disturbances	Avoid other cars, etc.
Set point	Minimum-time path
Cost function	Minimize circuit time
Receding horizon mechanism	Optimal path recalculated around the corner

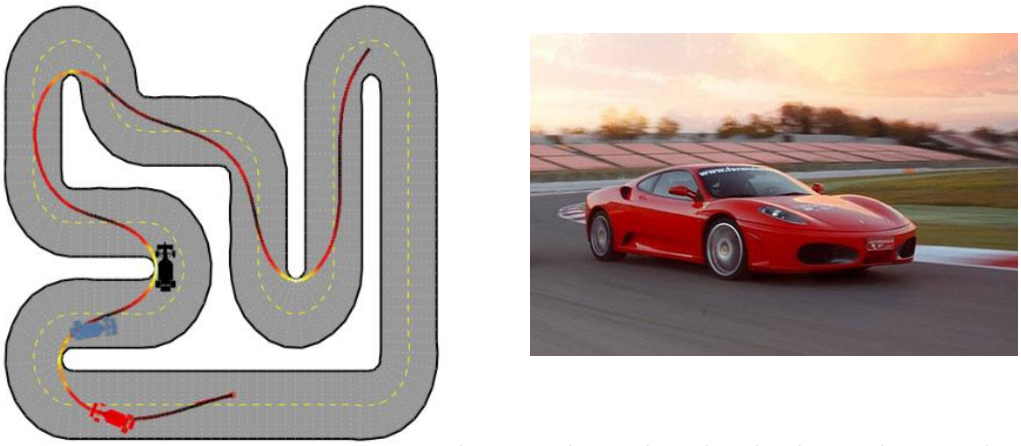


Fig. 4.10 Plan path and real trajectory in a car circuit

- **GPS System.**

Prediction model	How our vehicle moves on the map
Constraints	Drive on roads, respect one-way roads, respect traffic signals, etc.
Disturbances	Driver's inattention, road works, etc.
Set point	Desired location
Cost function	Minimize time Minimize distance, etc.
Receding horizon mechanism	Event based- (optimal-route recalculated when path is lost)

Fig. 4.11 Examples of a GPS System.

- **Chess.**

Prediction model	Each player has the ability to predict a finite number of opponent possible moves, given a fixed starting configuration of the chess board.
Prediction horizon	Good players think on long prediction horizon.
Optimization	The player must optimize his time by thinking certain number of future movements.
Apply first considered movement	Only one move is then carried out, leading to a new fixed configuration.
Receding horizon mechanism	The prediction process must be repeated for the new fixed configuration, which may differ from the previously predicted one.

Fig. 4.12 Chess playing.

4.6 MPC Advantages and disadvantages

Predictive controllers have been remarkably successful in the field of industry as well as in the research community. This is due to the properties of these control techniques, although there are some disadvantages.

Following advantages of the MPC must be noted:

- Straightforward formulation, flexible, opened and intuitive based on well understood concepts.
- Extremely configurable, changing model or specifications does not require complete redesign.
- Can add constraints over input, state, output and slew-rate variables.
  - Physical, safety, medium, economical, etc.
- It allows to treat with linear and not linear, monovariable and multivariable systems using the same formulation of the controller. Thus, predictive controllers can be used in a wide variety of processes: from simplest to more complex dynamics.
- Can be easily interpolated from the single-variable case to multi-variable case.
- Useful when a reference trajectory is available.
- Presents inherent compensation to dead-time and time delay phenomena.
- Control law answers are subjected to optimal criteria with well understood tuning parameters:
  - Prediction horizon
  - Optimization problem setup.

On the other hand, some of the disadvantages of this control technique are the following:

- Requires high level of accuracy in the determination of the system model in order to obtain a better closed-loop performance
- Requires an optimization algorithm, so it can only be implemented by a computer.
- Moreover, it requires a high computational cost for controller implementation (optimiser and model), which makes it difficult to apply to fast systems.
- Control law computation and controller tuning are not so trivial to perform (such as a PID law).
- It has not been so far, that the stability of the controllers was not guaranteed, especially in the case with restrictions. This made a heuristic adjustment of these controllers without a knowledge of how parameters could influence in the stability.
- Uncertainty is complex.

Observe that, predictive control is a very powerful technique that allows the formulation of controllers for complex and constrained systems.

This power has an associated price: the computational cost and the controller tuning.

Recent advances in the field of MPC provide a deeper understanding of these controllers, obtaining results that allow to relax these requirements. Thus, for example, general conditions have been established to guarantee stability, conditions under which the optimality of the controller can be relaxed guaranteeing its stability, and efficient algorithms for solving the problem have been developed.

4.7 MPC history and evolution

Model based predictive control is a method of advanced control originated in late seventies and early eighties in the process industries (oil refineries, chemical plants, etc.). A brief timeline of the main events of MPC is introduced [3]:

- First practical application of MPC algorithms appeared in industry (identified models):
    - IDCOM (Identification and command) – later MAC (Model algorithmic control) at ADERSA (Richalet, 1978).
    - DMC (Dynamic matrix control) at Shell (Cutler and Ramaker, 1980).
    - Many subsequent industrial developments (Setpoint, Honeywell, Pavillion, IPCOS, …).
- MPC algorithms were further developed in universities:
    - MUSMAR (Multistep multivariable adaptive regulator) – first state-space model formulation of MPC (Mosca, 1984).
    - EPSAC (Extended predictive self-adaptive control) – (De Keyser, 1985).
- Most notable generalizations, still used today:
    - Empirical (input/output) models:
        - GPC (Generalized predictive control) – (Clarke, 1987).
        - UPC (Unified predictive control) – (Soeterboek, 1992 – PhD student at TU/e, Electrical Engineering, Control Systems group).
    - First principle (state-space) models:
        - Very active area, many important contributions and many successful industrial applications. [4]

- The new model predictive control challenges are related with:
    - Nonlinear MPC.
        - Just need a computable model (simulation).
    - Hybrid MPC.
        - Discrete and parametric variables.
        - Combination of dynamics and discrete mode change.
        - Mixed-integer optimization (MILP, MIQP).
    - Engine control.
    - Large scale operation control problems.
        - Operations management (control of supply chain).
        - Campaign control.

Fig. 4.13 DMC plus, first powerful practical application of MPC by Shell company

4.8 Current MPC use in industry

The theory of MPC is well developed; most aspects, such as stability, nonlinearity, and robustness, have been discussed in the literature (see, e.g., (Bemporad & Morari, 1999) (Morari & Lee, 1999). Besides, MPC is very popular in the process control industry because the actual control objectives and operating constraints can be represented explicitly in the optimization problem that is solved at each control instant.

- In industry it has become the most used advanced control method [3]:
    - PID – 90%
    - Advanced – 10%
    - MPC – 9%
    - Other – 1%

- Majority of applications are in refining and petrochemicals. Chemical and pulp and paper are the next areas.

- Major developers/vendors of MPC include (IPCOS - Ton Backx):

Fig. 4.14 Today's important players in MPC industrial use

- In universities it has become one of the most prolific controller design methodologies in terms of both research grants and publications (more than 500 papers per year on MPC).

- Major universities and famous people in control are MPC developers: ETH Zurich (Manfred Morari), Imperial College London (David Mayne), Cambridge (Jan Maciejowski), Oxford (David Clarke), TU/e (Ton Backx), etc.

- Particularly suited for problems with:

    - Many inputs and outputs.
    - Constraints on inputs, outputs, states.
    - Varying objectives and limits on variables-(e.g.-because-of-faults).

- There are also some emerging MPC applications:

    - Vehicle path planning and control.
        - Nonlinear vehicle models.
        - With real world models.
        - Receding horizon preview.
    - Spacecraft rendezvous with space station.
    - Underwater vehicle guidance.
    - Missile guidance.

4.9 MPC control strategy applied on building

MPC principle of controlling house heating and cooling is based on the formulation of the building control as an optimization problem. This is archived by reacting on the heating/cooling actuators based on current measurements of temperatures and predictions of future disturbances (obtained from the weather forecast service).

The aim is mainly to design a control strategy that minimizes the energy consumption (or operational costs), while guaranteeing that all comfort requirements are met. The MPC will explicitly take into account the constraints of heating/cooling actuators and the temperature comfort limits.

- **System to be controlled**

Inputs:

- Weather (sun, temperature)
- Power to the heating

Outputs / States:

- Hot water temperature
- Room temperature

Constraints:

- Comfort criteria
- Equipment health aware

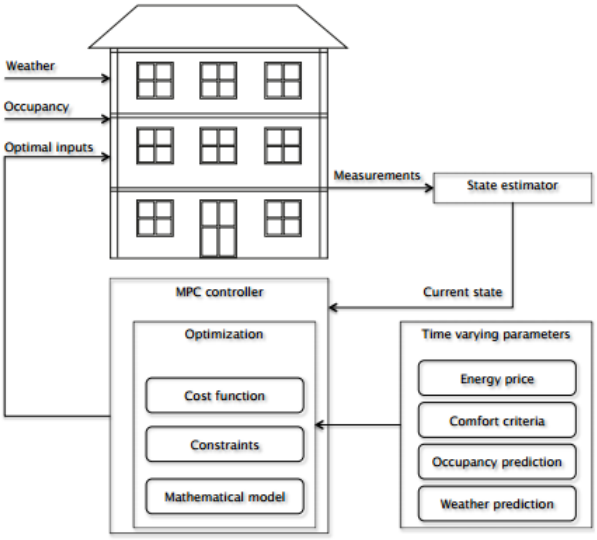


Fig. 4.15 Structure of model predictive controllers on a multizone building.

A predictive controller installed in a building is the element in charge of computing the following key points:

- Sequence of optimal inputs to the system (power to heating).
- Receive collected measurement in the building with sensors like temperatures and power heat. Also that allows to keep track of the history.
- Getting prognosis of future inputs from third parties:
  - Weather data (temperature, solar radiation)
  - Electricity price

- Run the following optimization problem:

$$
\begin{aligned}
&minimize_{u_0,\ldots,u_{N-1}} && \sum_{k=0}^{N-1} f_k\,(x_k, u_k) && \textit{Cost function}\\
&s.t && x_o = x && \textit{Current state}\\
& && x_{k+1} = f(x_k, u_k, w_k) && \textit{Dynamics – state update}\\
& && y_k = g(x_k, u_k, w_k) && \textit{Dynamics – system output}\\
& && (x_k, u_k)\ \epsilon\ X_k x\ U_k && \textit{Constraints}\\
& && (u_k): Inputs \quad (x_k): States \quad (y_k): Outputs &&
\end{aligned}
\tag{4.1}
$$

The building physics is formulated in a mathematical model that is used for the prediction of the future building behaviour according to the selected operation strategy and weather and occupancy forecasts. Using this information, the MPC controller computes the control action which minimizes a given cost function, while satisfying a set of constraints.

These constraints are most often:

1) Technical, e.g. they may limit the mass flow rate of a fan to its nominal value

2) Related to occupancy comfort, e.g. there may be a constraint on the building zone temperature(s).

The cost function is typically a weighted sum of the conflicting objectives of minimizing the energy cost and the thermal discomfort.

- **Control strategy**

In terms of building control, MPC strategy means that at the current control step, a heating or cooling plan is obtained for the next several hours or days, based on a weather forecast. Predictions of any other disturbances (e.g. internal gains), time-dependencies of the control costs (e.g. dynamic electricity prices), or of the constraints (e.g. thermal comfort range) can be readily included in the optimization.

The first step of the control plan is applied to the building, setting all the heating, cooling and ventilation elements, then the process moves one step forward and the procedure is repeated at the next time instant. This receding horizon approach is what introduces feedback into the system, since the new optimal control problem solved at the beginning of the next time interval will be a function of the new state at that point in time and hence of any disturbances that have acted on the building.

Figure 4.16 summarizes the basic principle of MPC for buildings. Time-varying parameters (i.e. the energy price, the comfort criteria, as well as predictions of weather and occupancy) are inputs to the MPC controller. One can see that the modelling and design effort consist of specifying a dynamic model of the building, as well as constraints of the control problem and a cost function that encapsulates the desired behaviours. At each sampling interval, these components are combined and converted into an optimization problem depending on the MPC framework chosen.

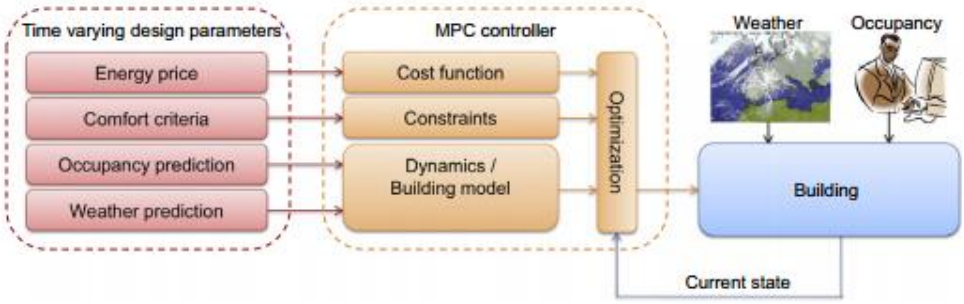


Fig. 4.16 Basic principle of MPC for buildings

5. LARGE SCALE BUILDINGS

5.1 Introduction

In recent years, there has been a growing concern to develop new control strategies that explore in buildings the concept of energy optimization resources in climate control systems. This heating, ventilation and air conditioning (HVAC) systems have a strong effect on the operating cost of building management and a strong environmental impact.

Approximately, about 40 % of total final energy consumption and more than half of the end energy [5] is consumed in heating, ventilation and air conditioning of all stances. Given this large share of energy consumed, improvement of buildings energy efficiency is crucial to ensure long-term energy security. Model Predictive Control (MPC) framework, due to its distinct advantages as reviewed in chapter 4, significantly outstands among other conventional methods applicable for the building control design.

This Chapter 5 is organized as follows. In Section 5.2 we introduce the problem of Large-scale systems (LSS) focused in building structures. In Section 5.3 we consider a great mathematical tool for the representation of models which is the state space. Then, two different configurations of HVAC are presented in Section 5.4 and finally a proper discussion of the common presumption of centrality in MPC which may lead to the innovation field of non-centralised control techniques. Finally, the current need of decomposition of control problem is briefly explained in Section 5.5.

5.2 Buildings energy problem as a Large Scale System

Modern innovations have increased the inner complexity of human implantations as there are more sized and advanced systems to regulate. The challenge of control of such vast infrastructures, usually referred as Large Scale Systems (LSS), rose since the last decades and its importance is becoming more and more crucial.

Large Scale Systems (LSS) are complex dynamical systems at service of everyone and in charge of industry, governments, and enterprises. The applications are wide. Typical examples of such systems are power networks, water networks, urban traffic networks, cooperating vehicles, digital cellular networks, flexible manufacturing networks, supply chains, complex structures in civil engineering, and many others.

Hence, the energy optimization problem of this project could be considered as a large-scale multi-variable control problem. Then, the global system is so-called Large Scale Building where the optimal control of HVAC system aims at providing the desired indoor comfort and

environment with least energy input under dynamic outdoor conditions and indoor loads.

It can be achieved by using suitable local controls of the sub-processes and optimal supervisory controls of the system. Also, it mainly depends on configuration of HVAC system and type of building.

For instance, let us consider three types of Large Scale Building such as a hotel, an office and an airport (check-in and waiting area. The building behaviour changes according to type of HVAC, system coupling between zones and occupancy schedule. The following table 5.1 illustrate comparison between zones.

	Type of HVAC	**Coupling between zone**	**Occupancy**
Office	Variable Air Volume Fan Coil Units	Weak	Almost fixed
Hotel	Fan Coil Units	Weak	Variable
Airport check-in and waiting area	Variable Air Volume Constant Air Volume	High coupling	Highly variable

Table 5.1 Comparison of different types of buildings.

According to the fact that MPC can manage a large number of variables in an easy way, this controller is also appropriate to be applied in these large-scale complex systems where there are many states, control outputs, and constraints.

Fig. 5.1 Considered types of buildings: Office, hotel and airport terminal

5.3 Introduction to state space representation

As previously discussed in § 4.2, one of the crucial contributors to the quality of the control is a well identified model which will be later on used for control in MPC algorithm.
The classical control theory and methods are based on a simple input-output description of the plant, usually expressed as a transfer function. These methods do not use any knowledge of the interior structure of the plant, and limit us to single-input single-output (SISO) systems.

However, modern control theory solves many of the limitations by using a much complete description of the plant dynamics. The so-called state-space description provide the dynamics as a set of coupled first-order differential equations in a set of internal variables known as state variables, together with a set of algebraic equations that combine the state variables into physical output variables.

5.3.1 System State

The concept of the state of a dynamic system refers to a minimum set of variables, known as state variables, that fully describe the system and its response to any given set of inputs.
In particular, a state-determined system model has the characteristic that:

A mathematical description of the system in terms of a minimum set of variables (called state variables) xi(t), i = 1,…,n, together with knowledge of those variables at an initial time t=t0 and the system inputs for time t ≥ t0, completely determine the behaviour of the system at any time t>t0 [6].

This definition proclaims that the dynamic behaviour of a state-determined system is completely characterized by the response of the set of n variables $x_i(t)$, where the number n is defined to be the order of the system.

There is no unique set of state variables that describe any given system; many different sets of variables may be selected to yield a complete system description. However, for a given system the order n is unique, and is independent of the particular set of state variables chosen. State variable descriptions of systems may be formulated in terms of physical and measurable variables, or in terms of variables that are not directly measurable. It is possible to mathematically transform one set of state variables to another; the important point is that any set of state variables must provide a complete description of the system. In this note we concentrate on a particular set of state variables that are based on energy storage variables in physical systems.

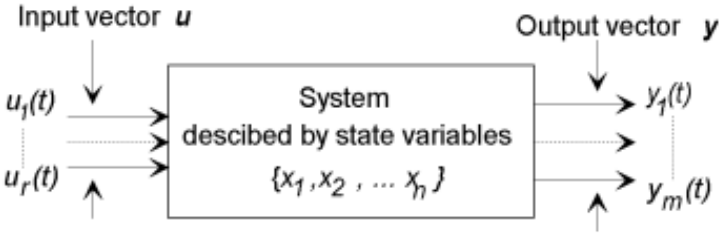


Fig. 5.2 State Space System representation

5.3.2 State Equations

In the standard form the mathematical description of the system is expressed as a set of n coupled first-order ordinary differential equations, known as the state equations, in which the time derivative of each state variable is expressed in terms of the state variables $x_1(t),...,x_n(t)$ and the system inputs $u_1(t),...,u_r(t)$.

In the general case the form of the n state equations is:

$$\begin{aligned} \dot{x_1} &= f_1(x, u, t) \\ \dot{x_2} &= f_2(x, u, t) \\ \vdots &= \vdots \\ \dot{x_n} &= f_n(x, u, t) \end{aligned} \qquad \begin{cases} \dot{x_1} = f_1(x_1, x_2, \cdots, x_n, u_1, u_2, \cdots, u_m, t) \\ \vdots \\ \dot{x_n} = f_1(x_1, x_2, \cdots, x_n, u_1, u_2, \cdots, u_m, t) \end{cases} \tag{5.1}$$

It is common to express the state equations in a vector form, in which the set of n state variables is written as a state vector $x(t)=[x_1(t), x_2(t),...,x_n(t)]T$, and the set of r inputs is written as an input vector $u(t)=[u_1(t), u_2(t),...,u_r(t)]T$. Each state variable is a time varying component of the column vector x(t).

This form of the state equations explicitly represents the basic elements contained in the definition of a state determined system. Given a set of initial conditions (the values of the xi at some time t_0) and the inputs for $t \geq t_0$, the state equations explicitly specify the derivatives of all state variables. The value of each state variable at some time Δt later may then be found by direct integration.

Moreover, one useful visualization of the state vector is to consider its time evolution in a space of n dimensions, where axes represent values of state variables $x_i(t)$, i = 1,...,n,. The system state at any instant may be interpreted as a point in an n-dimensional state space, and the dynamic state response x(t) can be interpreted as a path or trajectory traced out in the state space, where x(t) starts from the origin to coordinate point $x_1(t)$, $x_2(t)$; _ _ _; x_n (t).

In vector notation the set of n equations in Eqs. (1) may be written:

$$\dot{x} = f(x, u, t)$$

where f (x, u, t) is a vector function with n components fi (x, u, t). (5.2)

5.3.3 Output Equations

A system output is defined to be any system variable of interest. A description of a physical system in terms of a set of state variables does not necessarily include all of the variables of direct engineering interest.

An important property of the linear state equation description is that all system variables may be represented by a linear combination of the state variables xi and the system inputs u_i.

An arbitrary output variable in a system of order n with r inputs may be written:

$$y(t) = c_1x_1 + c_1x_2 + \cdots + c_nx_n + d_1u_1 + \cdots + d_ru_r \tag{5.3}$$

where the c_i and d_i are constants. If a total of *m* system variables are defined as outputs, the *m* such equations may be written as:

$$\begin{aligned} y_1 &= c_{11}x_1 + c_{12}x_2 + \cdots + c_{1n}x_n + d_{11}u_1 + \cdots + d_{1r}u_r \\ y_2 &= c_{21}x_1 + c_{22}x_2 + \cdots + c_{2n}x_n + d_{21}u_1 + \cdots + d_{2r}u_r \\ \vdots &= \vdots \\ y_m &= c_{m1}x_1 + c_{m2}x_2 + \cdots + c_{mn}x_n + d_{m1}u_1 + \cdots + d_{mr}u_r \end{aligned} \tag{5.4}$$

or in matrix form:

$$\begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_m \end{bmatrix} = \begin{bmatrix} c_{11} & c_{12} & \cdots & c_{1n} \\ c_{21} & c_{22} & \cdots & c_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ c_{m1} & c_{m2} & \cdots & c_{mn} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix} + \begin{bmatrix} d_{11} & \cdots & d_{1r} \\ d_{21} & \cdots & d_{2r} \\ \vdots & \ddots & \vdots \\ d_{m1} & \cdots & d_{mr} \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \\ \vdots \\ u_r \end{bmatrix} \tag{5.5}$$

The output equations, Eqs. (5), are commonly written in the compact form:

$$y = Cx + Du \tag{5.6}$$

where y is a column vector of the output variables $y_i(t)$, C is an m x n matrix of the constant coefficients c_{ij} that weight the state variables, and D is an m x r matrix of the constant coefficients d_{ij} that weight the system inputs. For many physical systems the matrix D is the null matrix, and the output equation reduces to a simple weighted combination of the state variables:

$$y = Cx \tag{5.7}$$

5.3.4 Linear systems

Definitely, the general case of the n state equations in Eqs. (5.1) allows us to represent a large number of systems. However, the mathematical treatment of equations is general unfavourable (possibly nonlinear systems) and not very productive.

On the other hand, if we limit ourselves to linear systems, all the tools of linear algebra are available:

$$\begin{cases} \dot{x}_1(t) = a_{11}(t)x_1(t) + \ldots + a_{1n}(t)x_n(t) + b_{11}(t)u_1(t) + \ldots + b_{1m}(t)u_m(t) \\ \vdots \\ \dot{x}_n(t) = a_{n1}(t)x_1(t) + \ldots + a_{nn}(t)x_n(t) + b_{n1}(t)u_1(t) + \ldots + b_{nm}(t)u_m(t) \end{cases}$$

$$\begin{cases} y_1(t) = c_{11}(t)x_1(t) + \cdots + c_{1n}(t)x_n(t) + d_{11}(t)u_1(t) + \cdots + d_{1m}(t)u_m(t) \\ \vdots \\ y_p(t) = c_{p1}(t)x_1(t) + \cdots + c_{pn}(t)x_n(t) + d_{11}(t)u_1(t) + \cdots + d_{pm}(t)u_m(t) \end{cases} \tag{5.8}$$

The linear system in Eq. (5.8) with p inputs, q outputs and n state variables is written in matrix form as:

$$\begin{aligned}\dot{\mathbf{x}}(t) &= \mathbf{A}(t)\mathbf{x}(t) + \mathbf{B}(t)\mathbf{u}(t)\\ \mathbf{y}(t) &= \mathbf{C}(t)\mathbf{x}(t) + \mathbf{D}(t)\mathbf{u}(t)\end{aligned} \tag{5.9}$$

where:

x *represents the state vector,* $x(t) \in \mathbb{R}^n$;
y *represents the output vector,* $y(t) \in \mathbb{R}^p$;
u *represents the input/control vector,* $u(t) \in \mathbb{R}^p$;
A *is the state matrix,* dim[A] = n x n;
B *is the input matrix,* dim[B] = n x p;
C *is the output matrix,* dim[C] = q x n;
D *is the feedthrough (or feedforward) matrix,* dim[D] = q x p

$$\mathbf{A}(t) = \begin{pmatrix} a_{11}(t) & a_{12}(t) & \cdots & a_{1n}(t) \\ a_{21}(t) & a_{22}(t) & \dots & a_{2n}(t) \\ \vdots & \vdots & \ddots & \vdots \\ a_{n1}(t) & a_{n2}(t) & \cdots & a_{nn}(t) \end{pmatrix}, \quad \mathbf{B}(t) = \begin{pmatrix} b_{11}(t) & \cdots & b_{1m}(t) \\ b_{21}(t) & \dots & b_{2m}(t) \\ \vdots & \ddots & \vdots \\ b_{n1}(t) & \cdots & b_{nm}(t) \end{pmatrix},$$

$$\mathbf{C}(t) = \begin{pmatrix} c_{11}(t) & c_{12}(t) & \cdots & c_{1n}(t) \\ c_{21}(t) & c_{22}(t) & \dots & c_{2n}(t) \\ \vdots & \vdots & \ddots & \vdots \\ c_{p1}(t) & c_{p2}(t) & \cdots & c_{pn}(t) \end{pmatrix}, \quad \mathbf{D}(t) = \begin{pmatrix} d_{11}(t) & \cdots & d_{1m}(t) \\ d_{21}(t) & \dots & d_{2m}(t) \\ \vdots & \ddots & \vdots \\ d_{p1}(t) & \cdots & d_{pm}(t) \end{pmatrix},$$

i

$$\mathbf{x}(t) = \begin{pmatrix} x_1(t) \\ x_2(t) \\ \vdots \\ x_n(t) \end{pmatrix}, \quad \mathbf{u}(t) = \begin{pmatrix} u_1(t) \\ u_2(t) \\ \vdots \\ u_m(t) \end{pmatrix}, \quad \mathbf{y}(t) = \begin{pmatrix} y_1(t) \\ y_2(t) \\ \vdots \\ y_p(t) \end{pmatrix}.$$

(5.10)

In this general formulation presented, time-variant (i.e. matrix elements can depend on time) is admitted in all matrices. However, in the common Linear Time-Invariant case, matrices will be time invariant. Time variable t can be continuous or discrete.

To sum up, depending on the assumptions taken, the state-space model representation can acquire the following forms:

System type	State-space model
Continuous time-invariant	$\dot{x}(t) = Ax(t) + Bu(t)$ $y(t) = Cx(t) + Du(t)$
Continuous time-variant	$\dot{x}(t) = A(t)x(t) + B(t)u(t)$ $y(t) = C(t)x(t) + D(t)u(t)$
Explicit discrete time-invariant	$x(k+1) = Ax(k) + Bu(k)$ $y(k) = Cx(k) + Du(k)$
Explicit discrete time-variant	$x(k+1) = A(k)x(k) + B(k)u(k)$ $y(k) = C(k)x(k) + D(k)u(k)$

Table 5.2 Different forms of State Space representation

5.3.5 State Space Model and Transfer Function model

To conclude this section a briefly comparison between the common Transfer Function model and the State Space presented:

- **Transfer Function Model**

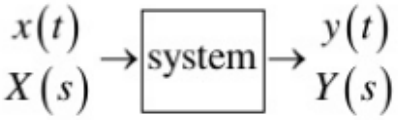


Fig. 5.3 System represented by transfer function model

Advantages

- Valid for unstable processes.
- Needs fewer parameters

Disadvantages

- Some processes may not be described sufficiently by a parametric model with a limited number of parameters.

- The structure of the process is fundamental for identification, especially the order of A and B.
- Transfer function is defined under zero initial conditions.
- Transfer function approach can be applied only to linear time invariant systems.
- It does not give any idea about the internal state of the system.
- It cannot be applied to multiple input multiple output systems.

➢ **State Space Model**

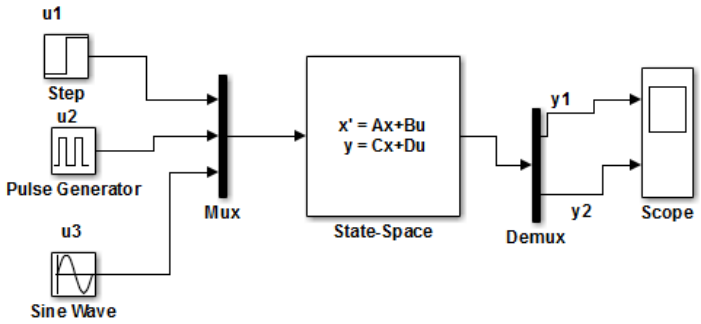


Fig. 5.4 Example of a system represented by State Space model

Advantages

- Multivariable processes can be presented in a straightforward manner.
- A large collection of modern control theory and analysis methods can be applied
- It can be applied to non-linear system.
- It can be applied to multiple input multiple output systems.
- Its gives idea about the internal state of the system.

Disadvantages

- The calculations may be complicated.
- Some processes may not be described sufficiently by a parametric model with a limited number of parameters.

5.4 Configurations of HVAC system

5.4.1 VAV based system

In a building, the air distribution system is responsible for maintaining appropriate levels of comfort in air quality, temperature, and moisture level. An example of such HVAC system is a Variable Air Volume (VAV), a type of air distribution system, that supplies not only one specific supply air stream of constant temperature to each space but also controls the temperatures within the building by changing the amount of air supplied to each space. [7]

VAV systems were developed to be more energy-efficient and to meet the varying heating and cooling needs of different building zones. The different parts of this system include the air-handling unit (AHU), the supply ductwork system, and the return/exhaust ductwork system.

- **Air Handling Unit**

The AHU is responsible for ventilating, mixing, filtering, conditioning, and supplying air throughout the building. It contains a mixer, a heating coil and a supply fan. This large air handler must have the ability to supply the Variable Air Volume boxes with a variable amount of air as the dampers in the Variable Air Volume system will modulate to different positions based on set point requirements.

The first section of the AHU is the mixing box. The mixing box is where outside air is brought in, supplied, and mixed with return air, which has been ducted from the building back to the AHU. Both the return and ventilation air are controlled by dampers, which help control the amount of unconditioned outside air brought into the system based upon minimum requirements set.

The next part of the air handling unit is the heat recovery unit. The heat recovery unit is used to transfer energy and moisture from the exhaust air leaving the building to the outside air entering the building.

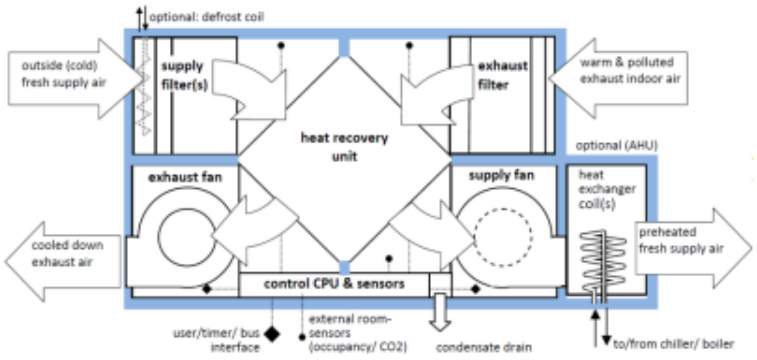


Fig. 5.5 Air Handling Unit of a VAV system.

After traveling through the heat recovery unit, the air passes through the condenser coils, heating coils, and cooling coils. The condenser coil is pumped with refrigerant so that as the air blows over this coil, any excess moisture will condense on the coil, run into the condensate drain, and exit the unit. The heating and cooling coils both accomplish the same job but are never in use at the same time. These coils are typically pumped full of steam and refrigerant respectively. A thermostat monitoring the outside air temperature operates both coils.

Once the air has been properly conditioned to the desired temperature and moisture, a supply air fan forces the air from the AHU into the supply ductwork.

- **The Supply Ductwork and VAV box**

After leaving the AHU, the conditioned supply air is ducted to the individual rooms for distribution. For each separate space within the building, the main supply duct will branch off to a VAV box.

A VAV box is another piece of mechanical equipment that consists of a motor-operated damper and a reheat coil. The damper in the VAV box is linked to a thermometer within the spaces to which air is supplied. This allows the thermometer to control the amount of air supplied in order to properly cool the space. One of the major issues with a VAV system arises when considering ventilation requirements for the building. According to the ventilation requirements, there is a minimum amount of fresh air that must be brought into the space. Whenever the amount of air needed to cool the space is less than the amount of air needed to meet ventilation requirements, a reheat coil is required. The reheat coil will slightly heat up the air being supplied to the space so that the minimum amount of ventilation air can still be supplied without over-cooling the space. Once the amount of air supplied to the space has been controlled and the temperature slightly adjusted if necessary, the supply air is ducted from the VAV box to diffusers, or air terminals, which distribute the air evenly throughout the space.

- **The Return/Exhaust Ductwork**

After the air has been supplied to the building, the excess air must be removed from the spaces as well. The return air system is ductwork that removes reusable or unpolluted air from the building and transports it back to the AHU. A fan is typically placed within the return ductwork to control the amount of air being removed from the spaces. The exhaust ductwork removes air from the building as well, but it removes 'polluted' air and forces air out of the building using an exhaust fan. 'Polluted' air is air that has been removed from bathrooms, kitchens, and other spaces where unwanted chemicals or fumes are released. This is where the heat recovery system is used. By pumping the conditioned exhaust air out of the building, energy is wasted. The heat recovery device typically is a slowly spinning wheel that can transfer energy and humidity from one source of air to another. By pumping the exhaust air through the energy recovery unit, the heat energy from the exhaust air can be transferred to the outside air initially entering the AHU. Energy can also be transferred from the outside air to the exhaust air on warmer days. By using this system, the overall energy consumption of a building can be reduced over time.

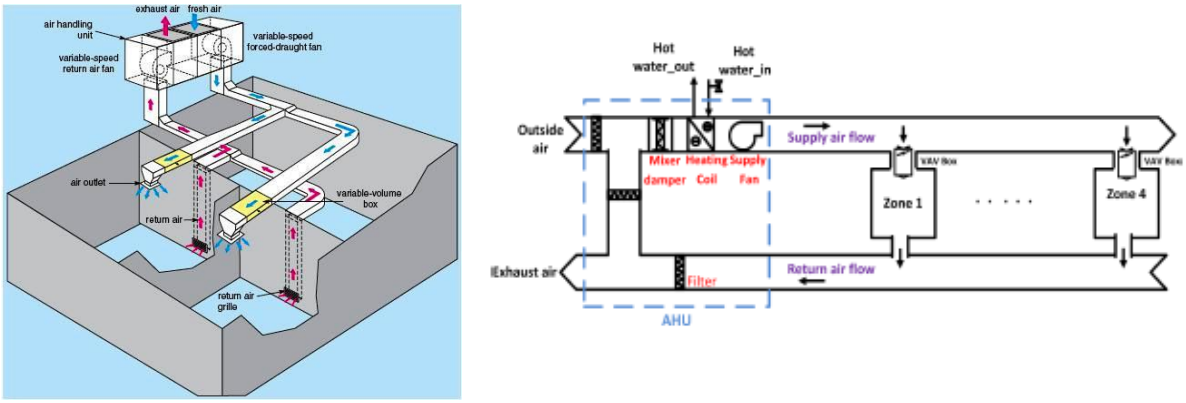


Fig. 5.5 Variable Volume Air (VAV) system.

5.4.2 FCU based system

A Fan Coil Unit (FCU) is a cooling system consisting of a heating or cooling coil and a fan. Warm air from the conditioned room is drawn into the fan coil unit, where it is cooled and dehumidified using chilled water before being supplied back into the conditioned room. The FCU units are equipped with filters that clean the air and hence reduce the level of airborne contamination within the air conditioned space. By using fan coil units, rooms can be cooled individually. [8]

The supply air temperature is controlled through the heat exchanger by controlling hot/cold water flow, depending on the temperature of space to which FCU serves. Hence unlike, AHU-VAV based system, zone model includes variables concerning heat exchanger and mixer.

- **Design and operation**

Fan Coil Unit design falls principally into two main types: blow through and draw through. As the names suggest, in the first type the fans are fitted such that they blow through the heat exchanger, and in the other type the fans are fitted after the coil such that they draw air through it. Draw through units are considered thermally superior, as ordinarily they make better use of the heat exchanger. However, they are more expensive, as they require a chassis to hold the fans whereas a blow-through unit typically consists of a set of fans bolted straight to a coil.

The coil receives hot or cold water from a central plant, and removes heat from or adds heat to the air through heat transfer. Traditionally fan coil units can contain their own internal thermostat, or can be wired to operate with a remote thermostat.

Fan coil units circulate hot or cold water through a coil in order to condition a space. The unit gets its hot or cold water from a central plant.

The equipment used can consist of machines used to remove heat such as a chiller or a cooling tower and equipment for adding heat to the building's water such as a boiler or a commercial water heater.

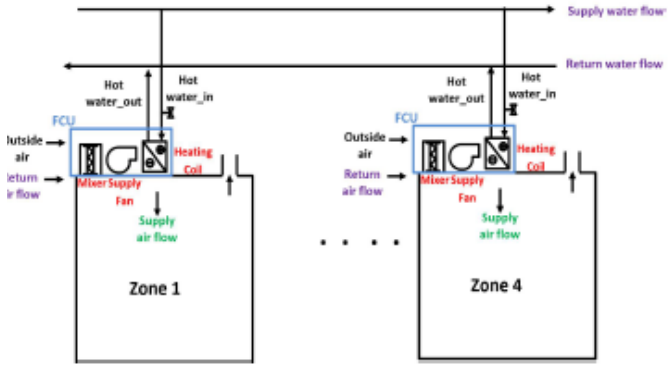


Fig. 5.6 Fan Coil Units (FCU) system.

5.5 Limitations of centralized control

Most of the procedures for controlling dynamical systems developed over the last decade rest on the common presupposition of centrality. Centrality means that all measurements and information available about the system must be collected in one location to estimate all states and to compute all control actions, which give the best possible performance. This information is divided into a *pre-information* about the dynamical model of the system available off-line, and a *post-information* about the system response gathered by different sensors.

Also, working with traditional MPC means that a global dynamical model of the system must be available for control design.

In this regard, there are some particular critical points to face when designing a centralised MPC and which have been object of research in the last years.

- Optimization problems become complicated. Since the number of decision variables and constraints in the optimization problem (for an established prediction horizon) is generally large, then the problem is computationally costly to solve.

- Coordination and communication between zones. There are some implementation issues related to communication availability and reliability. In the traditional MPC approach it is necessary to have the information of all system's states in a centralized scheme in order to compute the control outputs, which implies that it is necessary to send the decisions to all actuators to complete the closed-loop control**.** In this sense, costs associated to communication channels and operational concerns such as delays, noise or the loss of packets must be also considered.

- Maintenance. For large number of zones, fault in one zone can be propagated to other zones so maintenance is an important issue that requires cumbersome maintenance.

In order to mitigate computational burden, there are some immediate possible solutions such as simplify the corresponding model for prediction, reduce the prediction horizon, increase the sampling time to dispose of more time for computing the solution, or develop more sophisticated solvers in order to reduce computational time. However, this versatility of MPC might not be enough to solve this problem in certain design problems or in some complex systems.

5.5.1 Non-Centralized MPC Schemes for Large Scale Buildings

When considering large-scale systems, the previous presupposition of centrality usually fails to hold either because gathering all measurements in one location is not feasible, or because the computational needs of a centralised strategy are too demanding for a real-time implementation. This fact might lead to a lack of scalability. Subsequently, a model change would require the re-tuning of the centralised controller. Thus, the cost of setting up and maintaining the monolithic solution of the control problem is prohibitive.

In view of the above considerations, it is then natural to look for the Non-Centralized Model Predictive Control, in which the original large-size optimization problem is replaced by a number of smaller and easily tractable ones that work together in a possibly iterative and/or cooperative manner towards achieving a common wide system control objective. This type of predictive control is designed to operate in a decentralized or distributed way. With those techniques a set of local controllers (usually denoted as agents) are in charge of controlling partitions of the entire system.

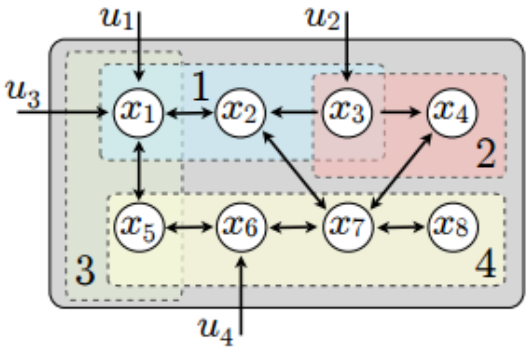


Fig. 5.7 Example of a global model decomposed into four submodels. Each coloured rectangle identifies the states belonging to the corresponding submodel.

The main difference between the two terms, decentralized or distributed MPC, depends on the type of information exchange among controllers:

- **Decentralized MPC:** Control agents take control decisions independently on each other. Information exchange (such as measurements and previous control decisions) is only allowed before and after the decision making process. There is no negotiation between agents during the decision process. The time needed to decide the control action is not affected by communication issues, such as network delays and loss of packets, thus easing control computation within the time deadline.

- **Distributed MPC:** In distributed control structures, it is assumed that some information is transmitted among the local control agents, so that each one of them has some knowledge on the behaviour of the others. The information transmitted typically consists of the future predicted control or state variables computed locally, so that any local regulator can predict the interaction effects over the considered prediction horizon.

Both MPC configurations, decentralized and distributed, are described more deeply as follows.

5.5.2 Decentralized MPC

The structure for the decentralized MPC divides the system as various subsystems. Each sub-system has an MPC that considers an optimization problem in which the variables are given by the actual sub-system parameters and the actual control action, whereas all the other elements in the system (i.e., control actions calculated by other sub-system MPC in the system or coupled states from other subsystem) are considered as external disturbances. Then at this structure, the local controllers do not exchange information each other. This approach allows the computation of the control actions with partial information, but it is not the optimal solution for the whole system (i.e., the solution is different from an MPC in which all the elements in the system are used to compute the control actions). When considering other elements of the system as disturbances, it is not taken into account the effects from other control actions that affect the behaviour of the total system, and the actual sub-system behaviour.

However, this decentralized controller is appropriate in cases where there is not coupled conditions among the sub-systems or when the influence of exogenous inputs is weak.

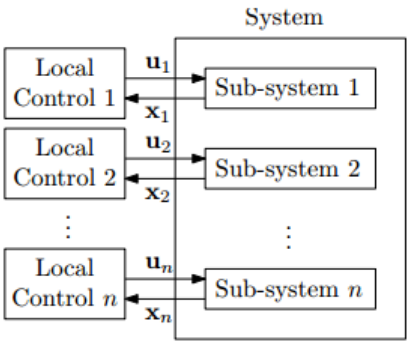


Fig. 5.8 Decentralized configuration. Local controllers dispose of the states of a subsystem but there is no exchange of information from others.

5.5.3 Decentralized MPC

The distributed control configuration is supposed to have a better performance with respect to the decentralized control configurations. This is because in the distributed schemes, the elements in the system can communicate partially each other, for which a coordination to achieve a better result is possible. In literature, the distributed structures have different classifications depending on how communication among elements is, and how the order of information requirements is (i.e., there are configurations in which the communication is made in both ways since sharing information is required, and others in which it is only required to have a one direction communication). Distributed schemes are also classified depending on the cost function that each element in the system takes into account (i.e., when in a local controller the global cost function is considered and cases where each local controller has a different cost function in comparison with other local controllers cost function).

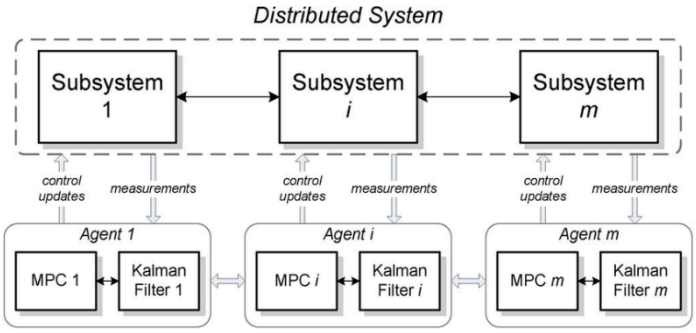


Fig. 5.9 Distributed configuration. Local controllers dispose of the states of a subsystem but does not have access to information from others. In this case, local controllers share partial information with other local controllers.

In a typical Distributed MPC (DMPC) framework the steps performed by the local controllers at each control instant are the following:

(i) measure local variables and update state estimates.
(ii) solve the local receding-horizon control problem.
(iii) apply the control signal for the current instant
(iv) exchange some information with other controllers.

5.5.4 Advantages and disadvantages of Non-Centralized MPC

Following benefits of non-centralized MPC must be taken into account:

- Reduced and parallel computations. Exist different elements computing control outputs, so the computational costs are divided by having different hardware processing the problem.

- Reduced communications. When it is not required to have centralized information, the reduction of communication channels is possible. It implies that the costs are reduced and the reliability improves, having less elements that could suffer a fault.

- Better maintenance. Local maintenance can be carried out by only stopping the corresponding local MPC controller. The remaining parts keep operating (possibly with reduced performance) in closed-loop with their local controllers, without the need of stopping the overall process as in case of centralized control.

- Possible re-design. A partial re-design of the process does not necessarily imply a complete re-design of the controller, as it would happen in case of centralized control.

Despite the benefits of non-centralised control architectures, they also have some drawbacks that must be noted:

- System contains coupled dynamics or coupled constraints. The non-centralizing task becomes challenging as it exists the difficulty to guarantee feasibility and it is possible the loss of performance in comparison with a single centralised controller. The solutions to these issues rely on the degree of interaction between the local subsystems and the coordination/communication mechanisms between their agents.

- Ensuring the asymptotic stability of the overall system. When all the controllers are involved in controlling the same large-scale process, it is important to determine conditions under which there are a set of appropriate local feedback control laws capable of stabilizing the entire system.

5.6 Decomposition aspect

When designing non-centralised controllers for large-scale buildings, there is a prior problem to be solved: the system decomposition into subsystems.

Usually the plant model is given as an atomic block of finite difference equations, thus a set of submodel is to be obtain through a process called “decomposition”. It consists on breaking the problem into small problems and solving them synchronous/asychronous way. There are class of techniques proposed in literature, to name a few, primal and dual decomposition, Dantzig-Wolfe decomposition and Augmented Lagrangian Decomposition etc.

The goal of the decomposition is double: first, each subsystem is much smaller than the overall problem (that is, each subproblem has far fewer decision variables and constraints than the centralized one), and second, each subsystem is coupled to only a few other subproblems (that is, it shares variables with only a limited number other subproblems).

5.6.1 Decomposition issues

Because of decentralizing the MPC problem may lead to a deterioration of the overall closed-loop performance due to the suboptimality of the resulting control actions, following key points must be noted,

- Cost function is convex (resource sharing).
- Equality constraints are linear or nonlinear (bilinear), depending on system dynamics.
- Equality constraints are complicated in nature.
- Complicated variables i.e. implicit variables in cost function and constraints.

In this work, it is assumed that the decomposition of the initial large-scale system into small-scale interacting subsystems is already given.

6. CENTRALIZED ECONOMIC MODEL PREDICTIVE CONTROL

6.1 Introduction

In this Chapter 6 we discuss a Centralised Economic Model Predictive Control structure. As explained in § 5.4.1, the predominant approach for controlling dynamical LSS buildings is based on the assumption of centrality where all information and control actions are ruled with a standard MPC controller designed for tracking economic operational set-points that are computed in a single optimisation problem.

In large non-residential and commercial buildings, the HVAC system must meet the varying needs of different spaces since different zones of a building may have different heating and cooling needs. In that respect, and facing that a centralised MPC fashion wants to be considered in this chapter, variable-air-volume (VAV) systems is a great HVAC configuration to meet these requirements. Furthermore, VAV configuration was developed to be more energy-efficient and to meet the varying heating and cooling needs of different building zones.

This Chapter 6 is organized as follows. In Section 6.2 we definitely introduce the HVAC configuration for this case. In Section 6.3 we consider the MPC formulation with the mathematical model, the economic cost function and the global building problem formulation.

6.2 Model Predictive Control formulation

6.2.1 HVAC Building configuration

The case-base under consideration in this work is properly explained in next Chapter 8 with the simulation results.

Each zone is equipped with:
- Temperature sensors.
- VAV box
- Return air plenum to recirculate the air from zones to: AHU (Air Handle Unit)

The variable-air-volume HVAC system works as follows in our building:

Element		Function
VAV terminal box		It receives primary air from the central air handling unit at the same constant temperature, known as the supply air-temperature.
Primary-air damper		It regulates the volume of hot or cold primary air delivered to the box according to the needs of the spaces. The damper position is controlled by a local PI type controller.
Air Handle Unit (AHU)	**Mixer**	It mixes the fresh air and the return air from all the zones
	Heating Coil	It is a water to air heat exchanger, which controls the temperature of supply air flow by varying hot water flow of constant temperature, supplied by a boiler.
	Supply fan	Fan that must vary its output in order to meet the needs of all VAV units. The speed of the central supply fan is consequently controlled to meet the changing demands of the building.

Table 6.1 All important elements forming the variable air volume HVAC system in a building

A mathematical model of the thermal behaviour of the zone and AHU, that is effectively used in control design, are properly explained at next subsections.

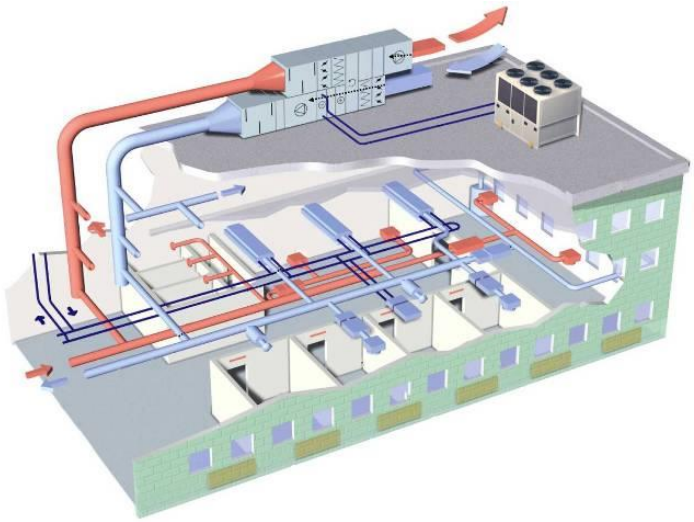


Fig. 6.1 All elements described at the above table in VAV configuration

6.2.2 Mathematical model of the building

1) Thermal zone model

Let n be the number of zones of the building. For each zone i, (i = 1, …n), denote the temperature of the zone by T_i, the mass flow rate at the output of the i:th VAV by $\dot{m}_i$ and the supply air temperature by T_s. Then, for the winter season, the first law of thermodynamics applied to each zone is

$$C_i\frac{dT_i}{dt} = \underbrace{\dot{m}_i c_p\left(T_s - T_i\right)}_{\substack{\text{Net amount of}\\ \text{Supply air flow}}} - \underbrace{\frac{1}{R_{ext_i}}\left(T_i - T_{oa}\right)}_{\substack{\text{Heat transfer to outside}\\ \text{environment}}} - \underbrace{\sum_{j=1,j\neq i}^{n}\frac{1}{R_{ij}}\left(T_i - T_j\right)}_{\substack{\text{Heat transfer to}\\ \text{adjacent zones}}} + \underbrace{q_i}_{\substack{\text{Heat flux due to}\\ \text{occupancy and}\\ \text{electronic devices etc..}}} \tag{6.1}$$

where:

- T_i *is the temperature of zone i;*
- T_s *is the supply air temperature;*
- T_{oa} *is the outside air temperature;*
- $\dot{m}_i$ *represents the mass flow rate at the output of the i:th VAV;*
- V_i *is the volume of zone i;*
- ρ *is the air density*;
- $C_i = \rho V i c_p$ *is the thermal capacitance of zone i*;
- q_i *represents the total internal heat gain*, *which is the cumulative heat flux due to occupants and electronic devices in the zone i;*
- *Rij = Rji is the thermal resistance between zone i and zone j*;
- $R_{ext,I}$ *is the thermal resistance between zone i and the exterior of the building*;

Clearly, the sensible heating load is the sum of all heat losses and internal heat gains. The heat loss for a zone i is due to the heat transfer from the zone i to adjacent zones j and to the outside environment.

Now, writing explicitly equation (6.1) for i = 1, …n, the model for the overall *n*-zones building is described by the system of *n* first-order differential equations written in matrix form as

$$\begin{pmatrix} C_1 & 0 & \cdots & 0 \\ 0 & C_2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & \cdots & 0 & C_n \end{pmatrix}\begin{pmatrix} \dot{T}_1 \\ \dot{T}_2 \\ \vdots \\ \dot{T}_n \end{pmatrix} = \begin{pmatrix} \frac{1}{R_{ext_1}} \\ \frac{1}{R_{ext_2}} \\ \vdots \\ \frac{1}{R_{ext_n}} \end{pmatrix} T_{oa} + \begin{pmatrix} -\alpha_{11} & \frac{1}{R_{12}} & \cdots & \frac{1}{R_{1n}} \\ \frac{1}{R_{21}} & -\alpha_{22} & \cdots & \frac{1}{R_{2n}} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{1}{R_{n1}} & \frac{1}{R_{n2}} & \cdots & -\alpha_{nn} \end{pmatrix}\begin{pmatrix} T_1 \\ T_2 \\ \vdots \\ T_n \end{pmatrix}$$

with

$$\alpha_{ii} = \frac{1}{R_{ext_i}} + \sum_{j=1,j\neq i}^{n}\frac{1}{R_{ij}}, \quad i = 1, ..., n.$$

$$+ \begin{pmatrix} \left(T_s - T_1\right) c_p \dot{m}_1 \\ \left(T_s - T_2\right) c_p \dot{m}_2 \\ \vdots \\ \left(T_s - T_n\right) c_p \dot{m}_n \end{pmatrix} + \begin{pmatrix} q_1 \\ q_2 \\ \vdots \\ q_n \end{pmatrix} \tag{6.2}$$

As the diagonal matrix in the left-hand side of (6.2) is not singular, the above equation can be written in the state space form,

$$x = \mathcal{A}x + (x - u_0 1_n)^t \mathcal{B}_u + \mathcal{G}_w + q \tag{6.3}$$

with $(.)^t$ as nomenclature of transposition,

$$x = [T_1 \quad T_2 \quad \dots \quad T_n]^t$$
$$u = [\dot{m}_1 \quad \dot{m}_2 \quad \dots \quad \dot{m}_n]^t$$
$$u_0 = T_s$$
$$w = T_{oa} \tag{6.4}$$

where:

- 1_n *is the n-dimensional vector with all entries equal to 1.*
- *A, B, G the structures of these matrices are straightforward from equation (6.2).*

For constant supply air temperature, which is typical to VAV based air-conditioning systems, the state equation is a bilinear controlled system,

$$\dot{x} = \mathcal{A}x + x^t \mathcal{B}_1 u + \mathcal{B}_2 u + \mathcal{G}_w + q \tag{6.5}$$

Finally, equation (6.5) is linearized around an operating point ($x^{(0)}$; $u^{(0)}$) and discretized with a sampling period h this leads to discrete-time system,

$$\begin{aligned} \tilde{x}(k+1) &= A\tilde{x}(k) + B_u\tilde{u}(k) + B_d\tilde{d}(k) \\ \tilde{y}(k) &= \tilde{x}(k) \end{aligned} \tag{6.6}$$

where:

- *A, B_u, B_d are the resulting discrete-time system matrices of appropriate dimensions;*
- *d =* [w ,q^t] *is the disturbance which accounts for outside temperature and internal gains q due to occupancy and electronic devices*;
- *x,u,d,w,q are the signals in equation(6) that denote currently small variations around their operating point values;*
- *y is the system output;*

2) Air Handling Unit

With reference to the following figure 6.3, let *Tr* denote the temperature of the return air flow rate at the input of the mixer. Then assuming that there is no leakage of mass flow rate in the duct, i.e.,

$$\dot{m}_a = (\sum_{i=1}^{n} \dot{m}_i) \tag{6.7}$$

the energy balance in the return duct reads as $\dot{m}_a T_r = \dot{m}_1 T_1 + \ldots + \dot{m}_n T_n$. This implies that the return temperature in the duct is completely determined by

$$T_r = \frac{\sum_{i=1}^{n} \dot{m_i} T_i}{\dot{m}_a} \tag{6.8}$$

Moreover, let T_m denote the temperature at the output of the mixer. The mixer mixes the return air at temperature T_r with fresh outdoor air at T_{oa}. Then, again, writing the energy conservation law for the mixer, we have

$$\dot{m}_r T_r + \dot{m}_{oa} T_{oa} = \dot{m}_a T_m \tag{6.9}$$

The conservation of mass at the inputs and output of the mixer implies $\dot{m}_r + \dot{m}_{oa} = \dot{m}_a$. The return mass air flow rate is a fraction δ ($0 \leq \delta \leq 1$) of the total mass airflow rate, i.e., $\dot{m}_r = \delta \dot{m}_a$, which implies that $\dot{m}_{oa} = (1-\delta)\dot{m}_a$. Then equation (6.9) reads as

$$T_m = \delta T_r + (1-\delta) T_{oa} = \delta \frac{\sum_{i=1}^{n} \dot{m_i} T_i}{\dot{m}_a} + (1-\delta) T_{oa} \tag{6.10}$$

T_m depends only on all zone temperatures T_i and on the outside temperature T_{oa}.

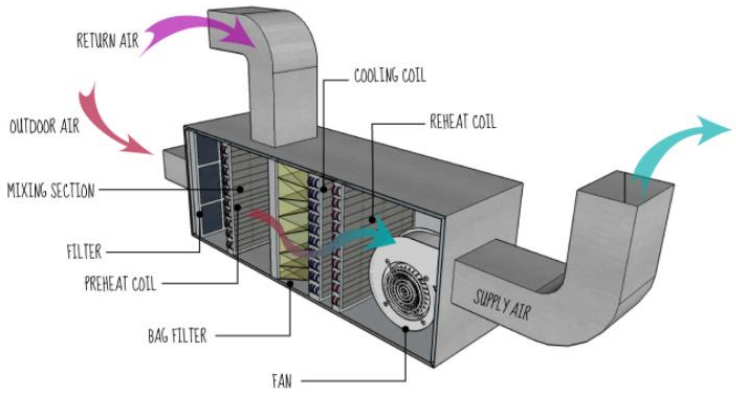


Fig. 6.3 Schematic of Air Handle Unit

6.2.3 Economic cost function

Economic cost function in the proposed model predictive control refers to the total cost of the energy consumed by the building components, mainly by the supply fan and a heating coils in the AHU. Let J be the total cost over a time interval of t_e,

$$J = \int_{t_e} (J_h + J_{fan})\, dt \tag{6.11}$$

where J_h and J_{fan} are the costs due to energy consumed by the heating coil and the supply fan in the AHU.

1) Energy cost at the heating coil

The power or heat transfer rate (Q_{coil}) required at the heating coil to deliver an air flow at temperature T_s is directly obtained from writing the energy conservation law,

$$\dot{Q}_{\substack{coil \\ AHU}} = \dot{m}_a c_p (T_s - T_m) \tag{6.12}$$

Then, the energy cost due to heating is simply given by

$$J_h = c_1 \dot{Q}_{\substack{coil \\ AHU}} \tag{6.13}$$

where c1 represents the related energy cost per kWh.

2) Energy cost delivered for the mass airflow

All the VAVs require a certain total mass airflow depending on each local (zone) heating load as per equation (6.7). This mass airflow is discharged by the power fan which is driven by a variable speed drive. The power fan characteristics is given by a cubic law, that is,

$$\dot{W}_{\substack{fan \\ AHU}} = \alpha \dot{m}_a^3 = \alpha (\textstyle\sum_{i=1}^{n} \dot{m}_i)^3 \tag{6.14}$$

With the above power characteristics, the cost the energy for a supply fan reads as follows,

$$J_{\substack{fan \\ AHU}} = c_2 \dot{W}_{\substack{fan \\ AHU}} \tag{6.15}$$

where c2 represents the related energy cost per kWh.

6.2.4 Formulation of Problem

The overall optimization problem for Model Predictive Control is formulated as below,

$$\begin{aligned} \Im(x,u) = minimize_{u,x} \quad & J(u,x) \\ s.t \quad & h(u,x,d) = 0 \\ & x_l \leq x \leq x_u \\ & u_l \leq u \leq u_u \end{aligned} \tag{6.16}$$

where:

- *J(u,x) is economic cost of HVAC system, explained in section 6.3.2;*
- *x,u are temperatures and supply airflow rates as states and inputs of system explained in section 6.3.1;*
- *xl, xu are lower and upper temperatures bounds which assures thermal comfort in the zones;*
- *ul, uu are lower and upper bounds on supply air ow actuators/temperatures*
- *d is disturbance vector;*

The cost function is convex while equality constraints are linear (system dynamics linearized at an operating point given in equation (6)), hence the optimization problem can be solved with existing optimization techniques e.g. interior point methods.

7. DISTRIBUTED ECONOMIC MODEL PREDICTIVE CONTROL

7.1 Introduction

With regard to the previous discussion introduced in section § 5.4, LSS Buildings are able to be either controlled by a single centralized (multi-variable) controller or by a set of decentralized control systems. Complexity in controlling these systems motivates the development of different architectures in the distributed control.

Typically, the design of the controllers of such a decentralized control structure does not account for the interactions between the subsystems explicitly. However, there is a desire to guarantee nominal stability, feasibility, optimality, reliability and maintainability for the control systems implemented in the plant. Distributed model predictive control (DMPC) methods are expected to contribute towards this aim because they can combine the optimality properties of centralized predictive controllers and the modularity and flexibility of decentralized control systems by means of communication, cooperation, or coordination.

In this chapter 7 we develop two techniques to solve the centralized problem thanks to decompose in a modular way the problem into small sized blocks. The first one, is based on optimality condition decomposition (OCD) and it is explained in Section 7.2. Secondly, in Section 7.3, it comes a new distributed model predictive control based on a distributed dynamic optimization algorithm. This technique is known as sensitivity-based coordination (S-DMPC) scheme. Finally, Section 7.5 explain the HVAC configuration for the non-centralised problems.

7.2 Optimal Conditions Decomposition in distributed model predictive control

7.2.1 Introduction

This section presents a partitioning technique based on decomposing the optimality conditions of the original problem, which is denominated optimality condition decomposition (OCD) [9]. It is based on the special structure of Karush-Kuhn-Tucker (KKT) system of the centralized large scale problem. The KKT system is carefully analysed and crucially modified in an efficient manner to obtain separable subsystems. These separable KKT subsystems imply the structure of the decomposed subproblems. The degree of modification in the original KKT system will define the convergence of the decomposed solution to the centralized solution.

7.2.2 Decomposition aspect

First of all, we consider a discrete time state space for a Large Scale System as,

$$x(k+1) = Ax(k) + B_u(k) \tag{7.1}$$

where:

- x represents the states of system, x(k) ∈ $\mathbb{R}^n$;
- u represents the input/control of system, u(t) ∈ $\mathbb{R}^m$;
- n is the number of the states of the system,
- m is the number of the inputs of the system,

Then, let us remind this proposed technique aims to give a solution of the optimal control problem for the overall system, Centralised Model Predictive Control (CMPC), introduced in § 6.3.3,

$$\begin{aligned} \Im(x,u) = minimize_{u,x} \quad & J(u,x) \\ s.t \quad & h(u,x,d) = 0 \\ & x_l \le x \le x_u \\ & u_l \le u \le u_u \end{aligned} \tag{7.2}$$

- J(x,u) is the overall cost function;
- h(u,x,d) are equality constraints that represents the dynamics of system in Eq.1;
- x_l, x_u, are lower and upper temperatures bounds (thermal comfort in the zones);
- u_l, u_u are lower and upper bounds on supply air ow actuators/temperatures;

1) Decomposition Structure

Once the main problem (7.2) is formulate, the decomposition process is introduced. First, we rewrite this problem into the equivalent general mathematical form to avoid cumbersome notations to simplify our system presentation,

$$\begin{aligned} F(z) = minimize_z \quad & f(z) \\ s.t \quad & h_j(z) = 0 \;\; j = 1, \dots, n_h \\ & z_{min} \le z \le z_{max} \end{aligned} \tag{7.3}$$

where z ∈ $\mathbb{R}^n$ is the vector of the optimization variables (x,u) and f is the convex twice differentiable objective function. The Lagrange function for the problem (7.3) with Lagrange multipliers λ (λ_1, …, λ_{nh}) reads as,

$$\mathcal{L}(z) = f(z) + \sum_{j=1}^{nh} \lambda_j h_j(z) \tag{7.4}$$

Moreover, if we consider *A* be the number of subproblems with partitioning the vector *z* as z_a (a = 1, 2, …, A) groups, z_a turn into the variables for each block k in which the original problem decomposes. Then, problem (7.3) can be written as:

$$\begin{aligned} &minimize \quad \sum_{a=1}^{A} f_a(z_a) \\ &\quad z_a(z) \;\; a = 1, \dots, A \end{aligned} \tag{7.5.1}$$

$$s.t \quad h(z_1, \dots, z_A) \leq 0 \tag{7.5.2}$$

$$g_a(z_a) \leq 0 \quad a = 1, \dots, A \tag{7.5.3}$$

The sets of equations (7.5.2-7.5.3) in that optimization problem represent both equality and inequality constraints.

It should be noted that these equations contain variables from different blocks and prevent each subproblem from being solved independently. This concept is known as overlapping and is an important issue to take into account when preparing decomposition of a main optimization problem. If these equations were removed from problem (7.5.1-7.5.3), the resulting problem could be trivially decomposed into one subproblem for each block. In this case, constraints would contain only variables belonging to block a for a = 1, . . ., A.

Considering that the optimal values of the Lagrange multipliers in problem (7.5.1-7.5.3) are known, the problem can be stated in an equivalent form as follows:

$$\begin{aligned} &minimize \quad \sum_{a=1}^{A} f_a(z_a) + \sum_{a=1}^{A} \lambda_a^T h_a(z_1, \dots, z_A) \\ &z_a(z) \;\; a = 1, \dots, A \end{aligned} \tag{7.6.1}$$

$$s.t \quad h_a(z_1, \dots, z_A) \leq 0 \quad a = 1, \dots, A \tag{7.6.2}$$

$$g_a(z_a) \leq 0 \quad a = 1, \dots, A \tag{7.6.3}$$

where constraints (7.5.2) have been separated in different blocks. Note that the way in which these constraints are distributed does not affect the solution of the problem, i.e., they can be distributed based on engineering insight. Given trial values to all variables and multipliers different than those in block a, problem (7.6.1) -(7.6.3) reduces to:

$$\mathcal{F}_a^t(z_a) = minimize \;\; k + \; f_a(z_a) + \sum_{\substack{b=1 \\ b \neq a}}^{A} \overline{\lambda_b}\, h_b(\widetilde{z_1}, \dots, z_A, \tilde{z}_{a+1}, \dots, \widetilde{z_A})$$

$$\begin{aligned} &s.t \quad h_a(z_a) \leq 0 \\ &z_{min} \leq z_a \leq z_{max} \end{aligned} \tag{7.7}$$

where

- $k = \sum_{\substack{b=1 \\ b \neq a}}^{A} f_b\,(\widetilde{z_b})$ is a constant
- $(\widetilde{z_1}, \dots, z_A, \tilde{z}_{a+1}, \dots, \widetilde{z_A})\ and\ \overline{\lambda_b} (b = 1, \dots, A; b \neq a)$ represents the optimal solution of other subproblems of previous (t -1) instants i.e. optimal solutions of $\mathcal{F}_a^t(z_a)\ (b = 1, \dots, A; b \neq a)$

Please note, inequality constraint can be transformed into equality constraints by adding slack variables.

This reduced problem (7.7) can be obtained for every block of the original problem. The proposed decomposition technique is actually based on the solutions of these reduced block-related problems. In fact, the selection of the suitable partition of the vector z into A groups and the decomposition of the set of equality constraints into h_a (a = 1,…, A) groups are important issues that the proposed algorithm try to face.

2) KKT system matrix

The notion of the decomposition is based on the partition of KKT system obtained from Lagrange function in equation (7.4).

KKT optimality conditions are first-order necessary conditions that must satisfy a solution of non-linear optimization problems with inequality constraints to be optimal. These conditions are a generalization of the Lagrange multiplier method for inequality constraints.

In general, KKT matrix can be derived by two ways,

KKT system matrix { Using developments in the primal dual interior point method
Gradient based approach

In this work only Primal-Dual procedure is applied, so it is presented briefly as follows.

Primal-Dual Approach (Interior point method)

Let us reconsider a general optimization problem from equation (7.3) and its Lagrange function from equation (7.4), where z (z1, …, z_A), and λ (λ_1, …, λ_{nh}) are primal and dual variables respectively. We assume the problem is strictly feasible then the KKT optimality conditions can be written as follows,

$$\nabla L(z^*, \lambda^*) = \nabla f(z^*) + \lambda^{*T} \nabla h(z^*) = 0 \qquad (7.8)$$

$$h_a(z^*) = 0 \qquad a = 1,2, \dots, A \qquad (7.9)$$

Above set of the KKT conditions can be solved for $(\tilde{n} + n_h)$ variables with respect to $(\tilde{n} + n_h)$ nonlinear equations. Interior point method solves the equations (7.6.1-7.6.3) and (7.7) by applying newton's method. This implies that this procedure seeks the search direction for both primal and dual variables, hence it is termed as Primal-Dual Interior Point Method. Nevertheless, the hierarchical procedure is described here to obtain KKT system as follows,

Let us define residual for the given time instant *t* as,

$$r_t(z, \lambda) = \begin{pmatrix} r_{dual} \\ r_{prim} \end{pmatrix} = \begin{pmatrix} \nabla f(z) + \lambda^T \nabla h(z) \\ h(z) \end{pmatrix} \tag{7.10}$$

where, r_{dual} is dual residual, r_{prim} is primal residual. Consider residual $r(y)$ in with search direction of $\nabla r(y)$, then Newton's step Δy is characterized by the linear equations as,

$$r(y + \Delta y) = r(y) + \nabla r(y) \Delta y \tag{7.11}$$

Then, if $y = (z, \lambda)$ and $\Delta y = (\Delta z, \Delta \lambda)$, the set of linear equations can be given as,

$$\begin{pmatrix} \nabla^2 f(z) + \lambda^T \nabla^2 h(z) & \nabla h(z) \\ \nabla h(z)^T & 0 \end{pmatrix} \begin{pmatrix} \Delta z \\ \Delta \lambda \end{pmatrix} = -\begin{pmatrix} r_{dual} \\ r_{prim} \end{pmatrix} \tag{7.12}$$

Please note that the values of primal and dual search directions depend on feasible initial values of z and λ. Newton's method can be extended in the case if the initial point is not feasible.

Furthermore, the matrix at the left hand side is termed as KKT system matrix which has a particular structure depending on the couplings in the dynamics of the system. The significance of the KKT matrix coefficients is concisely described in later key point of this subsection.

3) Significance of KKT system in the decomposition

KKT system matrix from Eq. (7.12) holds a crucial information structure about the system. It is symmetric and lower diagonal elements are zeros. Hence, two significant blocks are identified from which one block represent hessian of Lagrange function and other block represents the sensitivity of dynamic system.

- *Hessian of Lagrange Function*

  The upper triangular block from the KKT system (7.12) represents the separability of cost function with respect to the variable z and it is the hessian of lagrange function with respect to variable *z* i.e. $\nabla^2 f(z) + \lambda^T \nabla^2 h(z)$. If this block is in triangular form, then cost function is said to be separable.In that case, every sub-block represents the nature of cost function for the decomposed subproblems.

- *Sensitivity matrix*

The block matrix ($\nabla h(z)$) from the KKT system (12) represents the sensitivity of the system dynamics with respect to the vector z.

$$\begin{pmatrix} \nabla_{z1} h1(z) & \cdots & \nabla_{z\tilde{n}} h_{nh}(z) \\ \vdots & \ddots & \vdots \\ \nabla_{z1} h_{nh}(z) & \cdots & \nabla_{z\tilde{n}} h_{nh}(z) \end{pmatrix} \tag{7.13}$$

The off-diagonal coefficients in this block matrix represent the degree of coupling between variables z (z_1, …, z_n). The values of these coefficients have important role in the partitioning of constraints into *a* sets. If the coupling between the variables is weak, then those variables can be assigned to different groups.

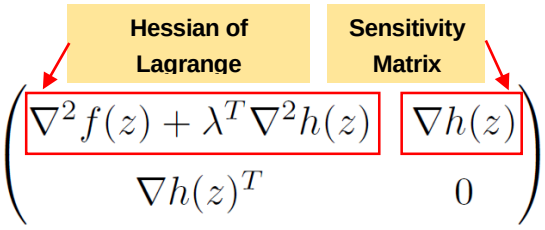


Fig. 7.1 Both blocks in KKT system matrix

Observe that, from the above key points, partitioning of the KKT system matrix achieves both the objectives of decomposition of the problem (7.3) on the control level as well as on the dynamics/constraints level. Hence the symmetric KKT matrix should be synthesized in the block diagonal form. In the literature, various methods describe the transformation of a symmetric matrix into block-diagonal matrix form.

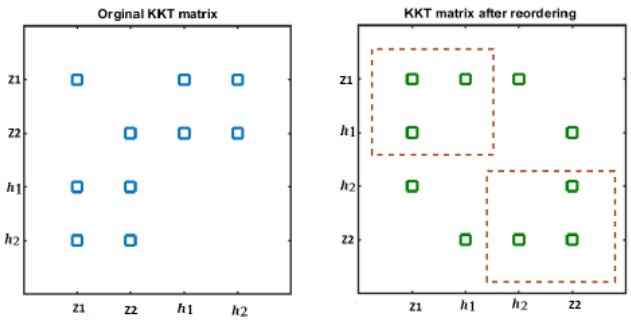


Fig. 7.2 Example of transformation of KKT matrix in "block-diagonal form"

In order to avoid the loss of information, it is advised to reorder the original KKT matrix into the block-diagonal structure. We use the Sparse Reverse Cuthill-McKee (CM) ordering algorithm. This algorithm permutes the rows/columns of sparse symmetric matrix to result in a narrow bandwidth band matrix. With this algorithm, it is possible to extract overlapping block diagonal matrices from the KKT system matrix. This algorithm is illustrated with an example as shown in Figure 7.3, where M is the sparse symmetric matrix and p is the permutation of rows/columns by sparse reverse Cuthill-McKee method.

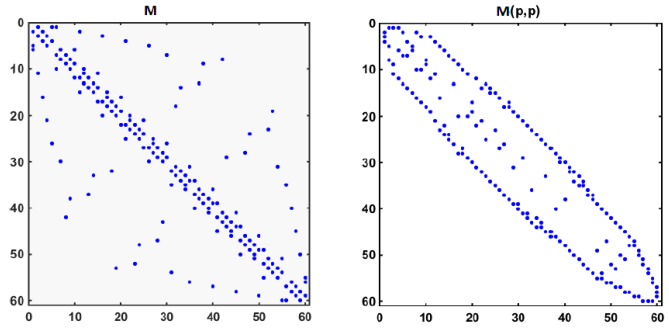


Fig. 7.3 Reordering KKT matrix into $\overline{KTT}$ with Reverse Cuthil-McKee Algoritm

Depending on the degree of coupling, a combination of decomposition method with CM algorithm results in more promising separable matrix structure.

In conclusion, we define the matrix $\overline{KKT}$ as the modified block diagonal matrix which can be decomposed into *a* smaller overlapping matrices.

The decomposed matrices explain the variable and constraints selection to obtain the subproblem as shown in the equation (7.5). The column in the decomposed matrix symbolizes the group of variables while the rows denote the constraints set to be considered in the respective subproblem.

4) Example

In order to understand the proposed procedure, we consider a simplified optimization problem. Please note, this is a case in which we have only two groups (n=2) of variables and additionally all constraints are equality ones.

$$\Im(z) = minimize_z \quad f(z)$$
$$s.t \quad h_i(z) = 0 \qquad i = 1,2 \ldots n \tag{7.14}$$

where:

- x is vector in $\mathbb{R}^n$;
- f and h are convex functions;

Let x be vector in $\mathbb{R}^2$ and equality conditions as $h_1(z_1, z_2) = 0 \;\; and \; h_2(z_1, z_2) = 0$:

$$\begin{aligned} \Im(z_1, z_2) = minimize_{z1,z2} \quad & f(z1, z2) \\ s.t \quad & h_1(z_1, z_2) = 0 \\ s.t \quad & h_2(z_1, z_2) = 0 \end{aligned} \tag{7.15}$$

Therefore, the decomposed problem with OCD method is written as,

$$\begin{aligned} \Im_1(z_1) = minimize_{z1,z2} \quad & f(z_1, \widetilde{z_2}) + \overline{\lambda_2}\, h_1(z_1, \widetilde{z_2}) \\ s.t \quad & h_1(z_1, \widetilde{z_2}) = 0 \end{aligned} \tag{7.16}$$

$$\begin{aligned} \Im_2(z_2) = minimize_{z1,z2} \quad & f(\widetilde{z_1}, z_2) + \overline{\lambda_1}\, h_2(\widetilde{z_1}, z_2) \\ s.t \quad & h_1(\widetilde{z_1}, z_2) = 0 \end{aligned} \tag{7.17}$$

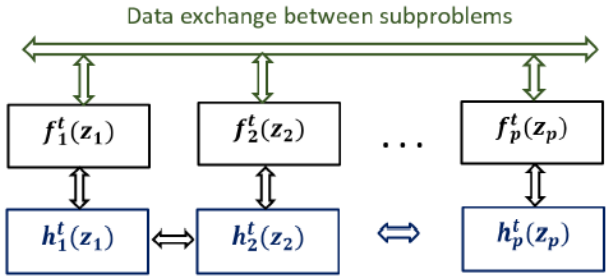


Fig. 7.4 Schematic of data exchange in case of overlapping decomposition

7.2.3 Convergence Properties

If the system in equation (7.3) is solved by computing optimal solutions for modular problems as shown in equation (7.7), the convergence criterion must be established. For following hypothesis, we assume that the problem in equation (7.3) holds the following properties,

1. The cost function f(z) is convex and twice differentiable;
2. The constraints functions h(z) (h1; …; h_A) are convex.

$$\|\lambda_{ev}(KKT) - \lambda_{ev}(\overline{KKT})\|^2 \leq \|\lambda_{ev}(\zeta)\|^2$$

where:

λ_{ev} is used represents as the symbolic representation to denote eigenvalues of the matrix;
ζ is the user defined tolerance matrix ;

Another way to prove convergence of the above method based on necessary condition of optimality (NCO). Let the Lagrange function for the subproblem from equation (7.5),

$$\mathcal{L}_a(z_a) = f_a(z_a, \widetilde{z_b}) + \overline{\lambda_b} h_b(z_a, \widetilde{z_b}) + \lambda_a h_a(z_a, \widetilde{z_b}) \quad \text{a} \neq \text{b} \tag{7.17}$$

The necessary condition of optimality (NCO) of the local problem is given by,

$$\frac{df_a(z_a, \widetilde{z_b})}{z_a} + \overline{\lambda_b} \frac{dh_b(z_a, \widetilde{z_b})}{z_a} + \lambda_a \frac{dh_a(z_a, \widetilde{z_b})}{z_a} = 0 \quad a \neq b \tag{7.18}$$

And considering NCOs for all *A* subproblems,

$$\sum_1^A \frac{df_a(z_a, \widetilde{z_b})}{z_a} + \overline{\lambda_b} \frac{dh_b(z_a, \widetilde{z_b})}{z_a} + \lambda_a \frac{dh_a(z_a, \widetilde{z_b})}{z_a} = 0 \quad a \neq b \tag{7.19}$$

The equation (??) is the same as NCO of the centralized problem. As centralized problem and subproblems are convex, optimization solution obtained from subproblems converges to the centralized solution.

If the constraints are linear i.e. when the second order differential matrix of constraint set *h* (*h1; … ; h*$_A$) shown in equation is a diagonal or null matrix, the optimal solution for the subproblems can be argued as same with the centralized solution.

$$\nabla^2 h = \begin{pmatrix} \nabla_{z1} h1(z) & \cdots & \nabla_{z\tilde{n}} h_{nh}(z) \\ \vdots & \ddots & \vdots \\ \nabla_{z1} h_{nh}(z) & \cdots & \nabla_{z\tilde{n}} h_{nh}(z) \end{pmatrix} \tag{7.20}$$

7.2.4 Algorithm OCD

A summary of the proposed algorithm is as follows.

Algorithm 1: Decomposition of Centralized Model Predictive Control (Optimality condition decomposition)	
Input Data	$\{h\ (h_1, \dots, h_A\ ; z^0\ (z_1^0, \dots, z_n^0); f(z)\} \rightarrow\ \mathcal{F}(z)$
Result	$\mathcal{F}_a^t(z_a)\ (\text{a} = 1, \dots, \text{A})$

Step 0: KKT matrix.

Formulate KKT system matrix for $\mathcal{F}(z)$.

Step 2: Partitioning of KKT system matrix.

a) Modify KKT system matrix into $\overline{KKT}$ using sparse reverse CM methods.
b) Until the condition $\|\lambda_{ev}(KKT) - \lambda_{ev}(\overline{KKT})\|^2 \leq \|\lambda_{ev}(\zeta)\|^2$ is satisfied.

Step 3: Overlapping / Nonoverlapping blocks

Identify overlapping / nonoverlapping separable blocks from modified $\overline{KKT}$ system matrix.

Step 4: Decomposed subproblems

Determine the decomposed subproblems $\mathcal{F}_a^t(z_a)$ $(\mathrm{a} = 1, \dots, \mathrm{A})$ based on the identified separable blocks.

Algorithm 2: The optimality condition decomposition	
Input Data	$\{\tilde{z}_a(\widetilde{z_1}, \dots, \widetilde{z_A}\,;\, \overline{\lambda_a}(\overline{\lambda_1}, \dots, \overline{\lambda_A}); Instant(t-1)\} \rightarrow \mathcal{F}_a^t(z_a)$ $(\mathrm{a} = 1, \dots, \mathrm{A})$
Result	$\widetilde{z_a}, \overline{\lambda_a}$ $(\mathrm{a} = 1, \dots, \mathrm{A})$; $Instant\ t$

Step 0: Initialization.

Each block (a = 1, . . . , A) initializes its variables and parameters, $\widetilde{z_a}, \overline{\lambda_a}$.

Step 1: Single iteration.

Each block (a = 1…, A) carries out one single iteration for its corresponding subproblem obtained in Step 4.

$$\mathcal{F}_a^t(z_a) = minimize_{z_a} \quad f_a(z_a) + \sum_{\substack{b=1 \\ b \neq a}}^{A} \overline{\lambda_b^T}\, h_b(\widetilde{z_1}, \dots, z_A, \tilde{z}_{a+1}, \dots, \widetilde{z_A})$$

$$s.t \qquad h_a(z_a) \leq 0$$
$$z_{min} \leq z_a \leq z_{max} \tag{7.21}$$

And obtain search directions Δz_a, $\Delta\lambda_a$.

The proposed approach has the advantage that convergence properties do not require to attain an optimal solution of the subproblems at each iteration of the algorithm. It is enough to perform a single iteration for each subproblem, and then to update variable values.

Step 2: Updating.

Each block (a = 1…, A) updates its variables and parameters.

$$\begin{aligned} \widetilde{z_a} + \Delta z_a &\rightarrow \widetilde{z_a} \\ \widetilde{\lambda_a} + \Delta\lambda_a &\rightarrow \widetilde{\lambda_a} \quad a = 1,2 \dots A \end{aligned} \tag{7.22}$$

Step 3: Stopping criterion.

The algorithm stops if variables do not change significantly in two consecutive iterations. Otherwise, it continues in Step 1.

7.3 Sensitivity-based coordination in distributed model predictive control

7.3.1 Introduction

In this section, we will introduce a new sensitivity-driven distributed model predictive control (S-DMPC) scheme [10], which is based on a novel distributed dynamic optimization algorithm motivated by so-called “goal-interaction operators”. We assume the objective function of the complete system to be separable.

Coordination and therefore overall optimality is achieved by means of a linear approximation of the objective functions of neighboring controllers within the objective function of each local controller. The objective functions of the subsystems are modified using information on the complete system to achieve optimality of the distributed control scheme.

Hence, coordination of the subsystem controllers is based on first order sensitivities. Each of the distributed controllers considers only a part of the full objective function and a reduced set of constraints and decision variables. We prove, that the distributed optimization method converges under given assumptions to the solution of the complete system. As for most of the distributed optimization methods, an iterative solution of the distributed optimal control problems is required.

7.3.2 Optimization control problem formulation

As the previous technique proposed, S-DMPC aims at the solution of the optimal control problem for the overall system, i.e.

$$\Im(x,u) = minimize_u\ \phi(u,x) = \sum_{i=1}^{N} \phi_i(u_i, x_i)$$

$$s.t \quad \phi_i = \frac{1}{2}\int_{t0}^{tf} x_i^T Q_i\, x_i + u_i^T R_i u_i \qquad i = 1,2 \ldots N$$

$$\dot{x} = Ax + Bu$$

$$x(t_0) = x_0$$

$$0 \le D(u,x) + e$$

where: (7.23)

$\phi_i(u,x)$ represent separable (or additive) quadratic objective function.
$Q_i\ and\ R_i$ are symmetric positive definite weighting matrices.
$[t0, tf]$ is the finite (receding) horizon.
$D = \langle D_1, \ldots, D_N \rangle$.
$e = \langle e_1, \ldots, e_N \rangle$.

In this case, objective function is formulated as a quadratic objective function because is one of the simplest form of non-linear programming. The open-loop optimal control problem (7.23) is transcribed into a Quadratic Programming problem (QP), that is the process of solving a special type of mathematical constrained optimization problem with a quadratic objective subject to linear constraints on these variables.

Following steps must be applied in the transcription into QP:

1. Discretize the input variables
2. Solve the state variables *x(k)* for the input parameters z , $z = \langle z_1, \dots, z_N \rangle$ represents the overall control inputs ,and the initial condition xo in discrete time.
3. Transform continuous-time cost function into discrete cost function.
4. Substitute *x(k)* in the discrete cost function.

This transcription is straightforward to result in the quadratic program (QP)

$$F(z) = minimize_z \sum_{i=1}^{N} \phi_i(z)$$

$$s.t \quad \phi_i(z) = \frac{1}{2} u^T A^i z + z^T B^i + C^i$$

$$c_i = D^{i\,T} z + E^i \geq 0$$

with

$$A^i = \begin{pmatrix} A^i_{11} & \cdots & A^i_{11} \\ \vdots & \ddots & \vdots \\ A^i_{N1} & \cdots & A^i_{NN} \end{pmatrix} \in \mathbb{R}^{npxnp}$$

$$B^i = \langle B^i_1, \dots, B^i_N \rangle \in \mathbb{R}^{np}$$

$$C^i \in \mathbb{R}$$

$$D^i = \langle D^i_1, \dots, D^i_N \rangle \in \mathbb{R}^{kixnp}$$

$$E^i \in \mathbb{R}^{ki}$$

$$\forall\, i = 1,2 \dots N \qquad (7.24)$$

where:

- k_i refers to the number of constraints associated with subsystem i.
- n_{pi} is the number of control parameters related to input vector ui $ki \leq n_{pi}$.
- n_p denotes the number of control parameters of the overall system $n_p = \sum_{i=1}^{N} n_{pi}$.
- A^i, B^i, C^i, D^i, E^i are computed from the matrices and vectors A, B, D, and e appearing in the optimal control problem (7.23)

7.3.3 Sensitivity-based coordination

The distributed solution of QP () requires its decomposition and the coordination of the resulting subproblems, which are described for a convex non-linear program (NLP).

Hence, a sensitivity-based coordination mechanism is proposed, where the overall objective ϕ is approximated in each of the local optimal control problems by a linearization of the contributions $\phi_j\ j \neq i$, of the neighbouring subsystems to the overall objective ϕ while the local quadratic objective function ϕ_i of the subsystem considered is retained.A convergence analysis is provided subsequently for the convex quadratic program.

Algorithm 3: The sensitivity-based coordination

For the sake of simplicity, we consider a more general Non-Linear Program:

$$F(z) = minimize_z \sum_{i=1}^{N} \phi_i(z)$$
$$s.t \quad c_i(z) \geq 0 \tag{7.25}$$

Step 0: Initialization.

Set k=0. Feasible parameters $z^{[0]}$ and an initial guess of the Lagrange parameters $\lambda^{[0]}$ are chosen.

Step 1: Parameters sent.

The control parameters $z_i^{[k]}$ and Lagrange parameters $\lambda_i^{[k]}$, $\forall \in \{1, \ldots, N\}$, are communicated to all local controllers.

Step 2: Solve the local optimization problems.

$$f(z_i) = minimize_z\ \phi_i{}^*(z)$$
$$s.t \quad c_i(z) \geq 0$$

$$with\ the\ strictly\ convex\ objective\ function$$

$$\phi_i{}^*(z) = \phi_i(z) + \left[\sum_{\substack{j=1 \\ j\neq i}}^{N} \frac{\partial \phi_j}{\partial z_i}\Big\|_{z^{[k]}}^{T} - \lambda_j^{[k]T} \frac{\partial c_j}{\partial z_i}\Big\|_{z^{[k]}}\right]\left(z_i - z_i^{[k]}\right) \tag{7.26}$$

where:

- k refers to the iteration index.
- c_j are the constraint functions related to system j.
- $\lambda_j^{[k]}$ are the Lagrange multipliers at iteration k related to the j-th NLP.
- $\frac{\partial \phi_j}{\partial z_i}$ are the first-order sensitivities of the objective function corresponding to subsystem j, $j \neq i$ with respect to the control parameters z_i.
- $\frac{\partial c_j}{\partial z_i}$ are the first-order sensitivities of the inequality constraints corresponding to subsystem j, $j \neq i$ with respect to the control parameters z_i.

This procedure is applied to obtain the local minimizers $z_i^{[k+1]}$and the Lagrange multipliers $\lambda_i^{[k+1]}$, ∀i ∈ {1, . . . , N}. Note, that the solution of the local optimization problems for $z_i^{[k+1]}$ assumes that all other control parameters are fixed at the previous iterates $z_j^{[k]}$, $\forall j \neq i$.

Step 3: Set k = k+1.

Go back to Step 1.

Step 4: Stop.

If $z^{[k]}$ satisfies some convergence criterion. There exist several possibilities for the choice of the stopping criterion in this step. In order to limit the deviation of the parameters $z^{[k]}$ from the optimal parameters z^*, a relative change of the parameters can be calculated as a simple measure for convergence, i.e.$\epsilon_{rel} = \frac{\| z^{[k]} - z^{[k-1]} \|_2}{\| z^{[k]} \|_2}$.

In conclusion, we are adding at each local controller in the objective function a "sensitive term" that is taking into account the coupling variables between subproblems. This strategy guaranties that the local optimization solutions separately converge to a global solution.

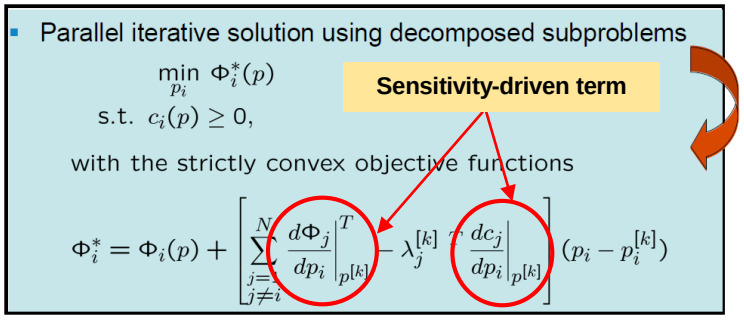


Fig. 7.5 Linearization term based on the contributions ϕ_j $j \neq i$, of the neighbouring subsystems to the overall objective ϕ.

7.3.4 Convergence Properties

First of all, some assumptions in centralized NLP (7.25) must be formulated:

- Cost functions ϕ_i are strictly convex.
- Constraint functions c_i are concave.

Then, the minimizer $z^{[k]}$, k→∞, of the local problems (7.26) and the minimizer z^*, of the centralized problem (7.25) are the same, i.e. lim k→∞$z^{[k]}$ = z^*.

Proof of optimality requires comparison of the NCO for the centralized problem and the decomposed problem:

Lagrange functions for the centralized problem

$$\mathcal{L}(z) = \sum_{j=1}^{N} (\phi_i(z) - \lambda_i^T c_i(z))$$

Lagrange functions for the centralized problem (7.27)

$$\mathcal{L}_i(z) = \phi_i(z) + \left[\sum_{\substack{j=1 \\ j \neq i}}^{N} \frac{\partial \phi_j}{\partial z_i}\bigg|_{z^{[k]}}^{T} - \lambda_j^{[k]T} \frac{\partial c_j}{\partial z_i}\bigg|_{z^{[k]}} \right] \left(z_i - z_i^{[k]} \right) - \lambda_i^T c_i(z) \tag{7.28}$$

At convergence, the necessary conditions of optimality (NCO) of the local problems are given by

$$0 = \frac{\partial \phi_i(z)}{\partial z_i} + \sum_{\substack{j=1 \\ j \neq i}}^{N} \left(\frac{\partial \phi_j(z)}{\partial z_i} - \frac{\partial c_j(z)^T}{\partial z_i} \lambda_j \right) - \frac{\partial c_i(z)^T}{\partial z_i} \lambda_i = \sum_{j=1}^{N} \left(\frac{\partial \phi_j(z)}{\partial z_i} - \frac{\partial c_j(z)^T}{\partial z_i} \lambda_j \right)$$

$$\begin{gathered} c_i(z) \geq 0 \\ \lambda_i \geq 0 \\ \lambda_i{}^T c_i(z) = 0 \\ \forall i \in (1, \ldots, N) \end{gathered} \tag{7.29}$$

which are the same as the NCO of the centralized problem. Since the NCO of the centralized and the local problems are the same and both problems have been assumed to be strictly convex, the minimizer computed from the iteration in Algorithm 3 is the same as the global minimizer of the centralized problem (7.25), provided the iterative solution of the former converges.

Since the convex NLP (7.26) and (7.25) generalize QP (7.23) and (7.24), if the latter are convex, convergence properties are also valid for the distributed solution of convex QP. For each of the local optimization problems (7.25), the overall objective function is considered, though all nonlocal contributions (i.e. those which result from other subsystems) are simplified by linear approximation.

In order to improve the convergence properties or even enforce convergence of the iteration, the objective function (7.25) can be extended to become:

$$\phi_i{}^{+} = \phi_i{}^{*} + \frac{1}{2}\left(z_i - z_i^{[k]}\right)^T \Omega^{\,i}\left(z_i - z_i^{[k]}\right) \qquad (1, \dots, N) \qquad (7.30)$$

$\boldsymbol{\Omega}^{\,\boldsymbol{i}} \in \mathbb{R}^{\boldsymbol{npixnpi}}$ is a symmetric positive definite matrix.

7.3.5 Algorithm S-DMPC

Here, the results are extended to the closed-loop S-DMPC formulation for the general QP (24). S-DMPC relies on the distributed optimization method discussed in the preceding section on a moving horizon $[t_0(h), t_f(h)]$, where *h* is the horizon index.

A summary of the proposed algorithm is as follows.

Algorithm 4: Closed-loop S-DMPC formulation	
Input Data	$\{z^{[0]}\,; \lambda^{[0]}\} \rightarrow \phi(z)$
Result	$\phi_i^{+}(z_i)(\mathrm{i} = 1, \dots, \mathrm{N}), z_i^{[k]}, \lambda_i^{[k]}$

Step 0: Initialization.

- Set h : = 0 and fix the initial system state $x(t_0(0))$.
- Transcribe the optimal control problem to compute $A^i, B^i(h), C^i(h), D^i, E^i(h)$; A^i *and* D^i do not depend on the initial state $x(h) = x(t_0(h))$ and need to be computed only once.
- Select initial parameters $z^{[0]}$ (h) and an estimate of the initial Lagrange multipliers $\lambda^{[0]}$ (h) and set k:=0.

Step 1: Send control parameters to local controllers.

Send the control parameters $z_i{}^{[0]}(h)$ and the Lagrange multipliers $\lambda_i{}^{[0]}(h)$, ∀i ∈ {1, . . . , N}, to the distributed controllers.

Step 2: Iterative process.

Solve the following QP on horizon $[t_0(h), t_f(h)]$ to obtain the minimizer $z_i{}^{[k+1]}$ and the Lagrange multiplier $\lambda_i{}^{[k+1]}$:

$$F_i(zi) = minimize_{zi}\ \phi_i^+$$

$$s.t \quad \phi_i(z) = \frac{1}{2}\tilde{z_i}^T A^i \tilde{z_i}^T + \tilde{z_i}^T B^i(h) + C^i(h)$$

$$+\left[\sum_{\substack{j=1 \\ j\neq i}}^{N}\left[A^i{}_{i1} \dots A^j{}_{iN}\right] z^{[k]} + D_i^j(h) - D^j{}_i(h)\lambda_j^{[k]}\right]\left(z_i - z_i^{[k]}\right) + \frac{1}{2}\left(z_i - z_i^{[k]}\right)^T \Omega^i \left(z_i - z_i^{[k]}\right)$$

$$c_i(\tilde{z_i}^{[k]}) = D^{i\,T}\tilde{z_i}^{[k]} + E^i(h) \geq 0$$

$$\forall\ i = 1,2 \dots N \qquad (7.31)$$

Step 3: Updating.

Increase k : = k + 1 and go back to 1.

Step 4: Stopping criterion.

Stop iteration, if $z^{[k]}$ satisfies some convergence criterion.

Step 5: Send calculated optimal controls to the system

- Apply the calculated optimal control inputs to the plant $u_{ij}(t)$.
- Set h : = h + 1, determine the new initial state $x(t_0(h))$either from measurements or from state estimation, and go back to 0.

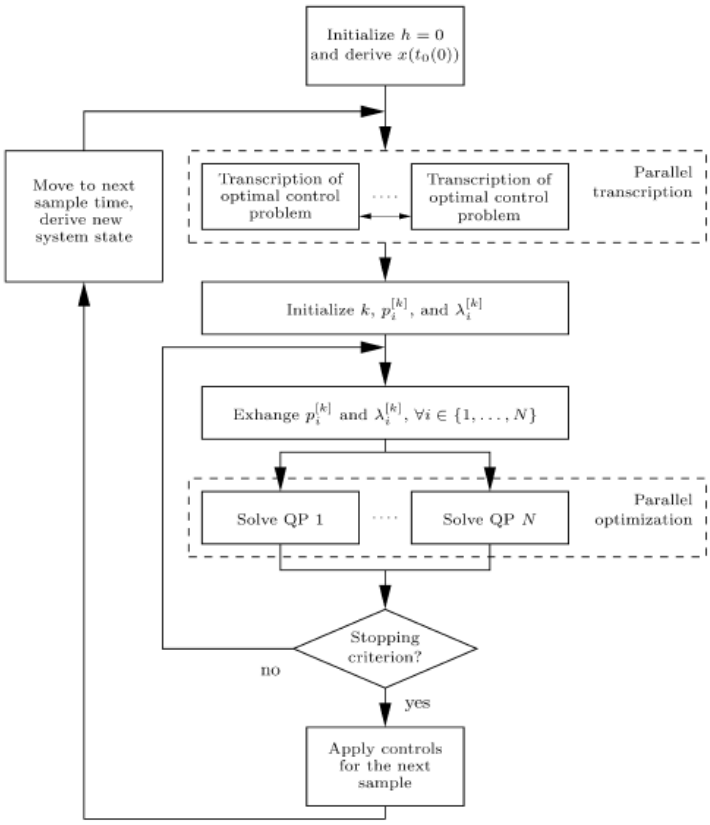


Fig. 7.6 Overview of the S-DMPC method.

7.4 Model Predictive Control formulation

7.4.1 HVAC Building configuration

The case-base under consideration in this work is properly explained in next Chapter 8 with the simulation results too. In this case, as we are analysing a distributed control, the FCU configuration can be applied to this type of system by considering every zone with its fan coil unit as a single subsystem.

Each zone is equipped with
- Temperature sensors.
- Fan Coil Unit
- Return air plenum

The FCU based heating, ventilation and air conditioning system works as follows in our building:

Element		Function
Fan Coil Unit (FCU)	**Mixer**	It mixes the fresh outside air with the return air flow from the plenum.
	Heating Coil	It is a water to air heat exchanger, which maintains the temperature of supply air flow at the required value by manipulating hot water flow from a boiler or heat pump.
	Supply fan	Fan maintains the constant supply air flow though the heating coil.
Return air plenum		It recirculates the fraction of the return air to FCU.
PID controller		It controls temperature by modulating supply air temperature in heating coil.

Table 6.1 All important elements that forms the fan coil units HVAC system in a building

Just a mathematical model of the thermal behaviour of each zone is formulated and it is properly explained at next subsection.

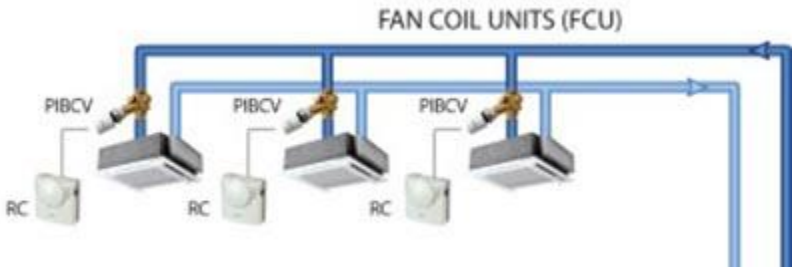


Fig. 6.1 Example of an implemented FCU configuration

7.4.2 Mathematical model of the building

1) Thermal zone model

For each zone i, (i = 1, …n), denote the temperature of the zone by T_i, the mass flow rate at the output of the i[th] FCU by $\dot{m}_i$ and the supply air temperature by T_{si}. Then, the first law of thermodynamics applied to each zone is

$$C_i \frac{dT_i}{dt} = \dot{m}_i c_p \left(T_{s_i} - T_i\right) - \frac{1}{R_{ext_i}} \left(T_i - T_{oa}\right) - \sum_{j=1, j\neq i}^{7} \frac{1}{R_{ij}} \left(T_i - T_j\right) + q_i \tag{7.32}$$

where:

- T_i *is the temperature of zone i;*
- T_{si} *is the supply air temperature;*
- T_{oa} *is the outside air temperature;*
- $\dot{m}_i$ *represents the mass flow rate at the output of the i:th VAV;*
- V_i *is the volume of zone i;*
- ρ *is the air density*;
- $C_i = \rho V i c_p$ *is the thermal capacitance of zone i*;
- q_i *represents the total internal heat gain*, *which is the cumulative heat flux due to occupants and electronic devices in the zone i;*
- *Rij = Rji is the thermal resistance between zone i and zone j*;
- $R_{ext,I}$ *is the thermal resistance between zone i and the exterior of the building*;

Equation (7.32) is written in general form as,

$$\dot{x}_i = \mathcal{A}_i x_i + \sum_{\substack{j=1 \\ j \neq i}}^{N} \mathcal{A}_{ij} x_j + \mathcal{B}_i u_i + \mathcal{G}_i Toa + q_i \quad (7.33)$$

Now, continuous-time model (7.33) for i = 1, …n, is discretized with a sampling period h and linearized around an operating point (x_i (0); u_i (0)) which leads to discrete-time system,

$$x_i(k+1) = A_i x_i(k) + A_{ij} x_j(k) + B_i u_i(k) + \mathcal{G}_i d(k) \quad (7.33)$$

here:

- *A_i, A_{ij} B_i, G_i are the resulting discrete-time system matrices of appropriate dimensions;*
- *d_i = [w ,q_i]^T is the disturbance which accounts for outside temperature and internal gains q due to occupancy and electronic devices*;
- *x_i,x_j,u_i,d_i, are the signals in equation(7.33) that denote currently small variations around their operating point values;*

7.4.3 Economic cost function

Economic cost function in the proposed model predictive control refers to the total cost of the energy consumed by the building components, mainly by the supply fan and a heating coils in the FCUs. Let J_i be the total cost over a time interval of [t_0,t_f] for i^{th} zone,

$$J_i = \int_{tf}^{t0} (J_{hi} + J_{fan\,i}) dt \quad (7.34)$$

where J_{hi} and J_{fani} are the costs due to energy consumed by the heating coil and the supply fan in the FCU of the i^{th} zone.

1) Energy cost at the heating coil

The power or heat transfer rate ($\dot{Q}_{coil\,i}$) required at the heating coil to deliver an air flow at temperature T_s is directly obtained from writing the energy conservation law,

$$\dot{Q}_{coil\,i} = \dot{m}_i c_p (T_{s_i} - T_{m_i}) \quad (7.35)$$

Then, the energy cost due to heating is simply given by

$$J_{h_i} = c_1 \dot{Q}_{coil\,i} \quad (7.36)$$

where c1 represents the related energy cost per kWh

2) Energy cost delivered for the mass airflow

The FCUs require a certain total mass airflow depending on each local (zone) heating load.

This mass airflow is discharged by the power fan which is driven by a variable speed drive. The power fan characteristics is given by a cubic law, that is,

$$\dot{W}_{fan_i} = \alpha \dot{m}_i^3 \tag{7.37}$$

With the above power characteristics, the cost the energy for a supply fan reads for the i^{th} zone-FCU as follows,

$$J_{fan_i} = c_2 \dot{W}_{fan_i} \tag{7.38}$$

where c2 represents the related energy cost per kWh

7.4.4 Formulation of Problem

The overall optimization problem for Model Predictive Control is formulated as below,

$$\begin{gathered} \Im(x,u) = minimize_{u,x} \quad V(u,x) \\ s.t \;\; x_i(k+1) = A_i x_i(k) + A_{ij} x_j(k) + B_i u_i(k) + \mathcal{G}_i d(k) \\ x_l \le x \le x_u \\ u_l \le u \le u_u \end{gathered} \tag{7.39}$$

where:

- *V(u,x) is the total power consumption in the operational stage of the building;*
- *x_i,x_j,u_i are temperatures and supply airflow rates as states and inputs of system;*
- *xl, xu are lower and upper temperatures bounds which assures thermal comfort in the zones;*
- *ul, uu are lower and upper bounds on supply air ow actuators/temperatures*
- *d is disturbance vector;*

8. SIMULATION RESULTS AND COMPARATIVE ANALYSIS

8.1 Introduction

This chapter aims to compare the two distributed model predictive control approaches with the centralised solution in a LSS Building. To verify the performance of the controllers, we simulated the combination of controller and plant. In this work MATLAB software is used with the modelling language YALMIP, optimizer solver Gurobi] and the semidefinite-quadratic linear programing solver SDPT3.

However, the main issue of this work is that both distributed model approaches are already formulated but in different type of building. This is because we take as a point of departure the previous work done by Tejaswinee Darure [11], which is part of Energy in Time project funded by the European Union. Table 8.1 give evidence of this inconvenient that disables a direct comparative.

	Case 1	**Case 2**
Type of building	School of 8 zones	Office of 6 zones
Coupling between zone	Coupling of u	Coupling of states x
Initial MPC available formulation	**S-DMPC**	**CMPC** **OCD-DMPC**

Table 8.1 Initial situation of DMPC formulated but in different type of buildings.

Joined to the above problem, a solution was considered. It consists on the reformulation of the MPC formulation of one case into the other that is missing. For example, an option was to reformulate the OCD-DMPC to S-DMPC in order to have both approaches at the same type of building.

Next, all model predictive control approaches starts from a detailed dynamic non-linear building model, which is a physical modelling using heat transfer equations. These equations are describing the thermal behaviour of the building envelope and are non-linear. Hence, equations are linearized around an operating point to obtain a state space controller model.

In conclusion, in this chapter we present simulation results for the 8 and 6 zones building of available MPC formulations.

This Chapter 8 is organized as follows. In Section 8.2 we introduce the benchmark building description of both buildings. Then, as an introduction to all MPC configurations we want to implement an illustrative example is presented in Section 8.3 where all centralized a non-centralized algorithm are formulated. In Section 8.4 we analyse simulation result for the centralized solution. Finally, the simulation results for the two different configurations of DMPC are presented in Section 8.5.

8.2 Benchmark Building Description

8.2.1 Eight zones building

The first case-base under consideration in this section consist on a school building of 8 zones. It is distributed in four central zones each of size 6m x 6m flanked of two right and left wings each of one is composed of two zones of size 12m x 12m. It has a total area of 720 m^2. Figures 8.1-8.2 show the layout of the building.

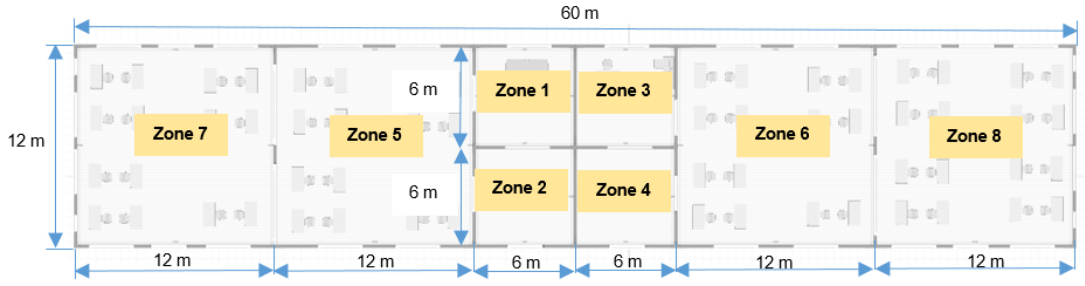


Fig. 8.1 Distribution and sizes of all zones in the building layout

Fig. 8.2. General plant view of this school of eight zones

8.2.2 Six zones building

The second case-base under consideration in this section consist on an office building of 6 zones. The type of occupancy is that of an office building where the working time is from 8:00 to 18:00 with a midday break of two hours starting at 12:00. It is distributed in four central zones each of size 6m x 6m flanked of two zones of size 12m x 12m. It has a total area of 432 m^2. Figures 8.3-8.4 show the layout of the building.

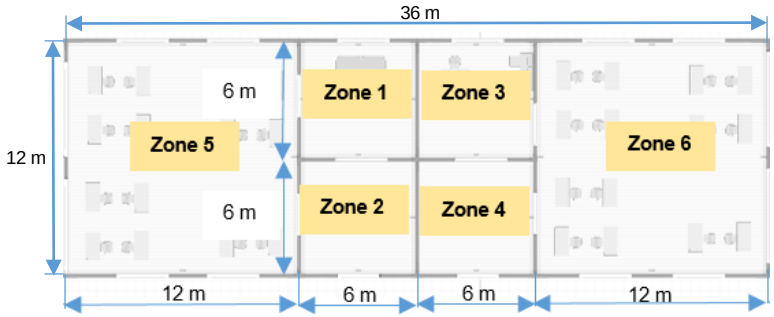


Fig. 8.3 Distribution and sizes of all zones in the building layout

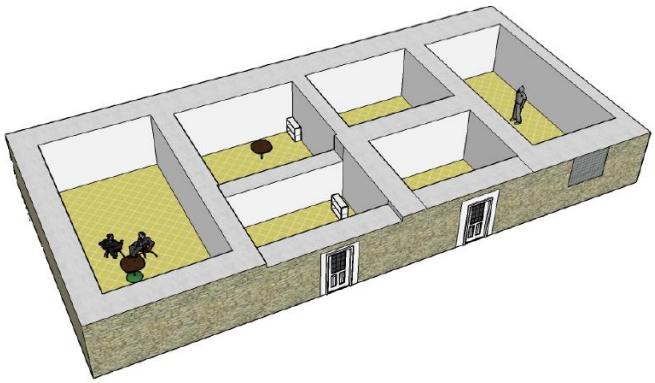


Fig. 8.4 A general view of this office of eight zones

8.3 Simulation study: Illustrative example

In this section we introduce an example obtained from the book of A.Conejo[9]. In this book only the OCD approach is presented. A deeper analysis is considered by also computing the centralised solution and the sensitivity-based coordination approach too.

8.3.1 Centralized solution

The problem to be solved is

$$\Im(x_1, x_{2,} y_1, y_2) = minimize_{x1,x2,y1.y2} \quad x_1^2 + x_2^2 + y_1^2 + y_2^2$$

$$s.t \quad h_1(x_1, x_{2,} y_1, y_2) = 4x_1 + y_2 - 1 = 0$$
$$h_2(x_1, x_{2,} y_1, y_2) = x_1 + 4y_2 - 1 = 0 \tag{8.1}$$

The variables, constraint and function vectors are,

$$x = \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} \qquad y = \begin{pmatrix} y_1 \\ y_2 \end{pmatrix} \tag{8.2}$$

$$h(x, y) = \begin{pmatrix} 4x_1 + y_2 - 1 \\ x_1 + 4y_2 - 1 \end{pmatrix} \tag{8.3}$$

$$f(x, y) = \begin{pmatrix} 4x_1 + y_2 - 1 \\ x_1 + 4y_2 - 1 \end{pmatrix} \tag{8.4}$$

The Lagrangian function is written as,

$$\mathcal{L}_{x,y}(x_1, x_{2,} y_1, y_{2,} \lambda_1, \lambda_2) = x_1^2 + x_2^2 + y_1^2 + y_2^2 + \lambda_1 (4x_1 + y_2 - 1) + \lambda_2 (x_1 + 4y_2 - 1) \tag{8.5}$$

Finally, the solution for this problem is

$$\begin{aligned}
&\frac{\partial \mathcal{L}_{x,y}}{\partial x_1} = \mathbf{2x_1 + 4\lambda_1 + \lambda_2 = 0} \\
&\frac{\partial \mathcal{L}_{x,y}}{\partial x_2} = \mathbf{2x_2 = 0} \\
&\frac{\partial \mathcal{L}_{x,y}}{\partial y_1} = \mathbf{2y_1 = 0} \\
&\frac{\partial \mathcal{L}_{x,y}}{\partial y_2} = \mathbf{2y_2} + \lambda_1 + \mathbf{4\lambda_2 = 0} \\
&\frac{\partial \mathcal{L}_{x,y}}{\partial \lambda_1} = \mathbf{4x_1 + y_2 - 1 = 0} \\
&\frac{\partial \mathcal{L}_{x,y}}{\partial \lambda_2} = \mathbf{x_1 + 4y_2 - 1 = 0}
\end{aligned} \qquad x^* = \begin{pmatrix} 0.2 \\ 0.0 \end{pmatrix} \quad y^* = \begin{pmatrix} 0.0 \\ 0.2 \end{pmatrix} \quad \lambda^* = \begin{pmatrix} -0.08 \\ -0.08 \end{pmatrix} \tag{8.6}$$

8.3.2 Optimal Conditions Decomposition (OCD) solution

Using the proposed methodology in § 7.2, the subproblems to be solved in Step 6 of the OCD decomposition algorithm 1 are, respectively,

$$\Im_1(x_1, x_{2,}) = minimize_{x1,x2} \quad x_1^2 + x_2^2 + \overline{\lambda_2}\,(x_1 + 4\overline{y_2} - 1)$$

$$s.t \qquad 4x_1 + \overline{y_2} - 1 = 0 \tag{8.7}$$

$$\Im_2(y_1, y_2) = minimize_{y1.y2} \; y_1^2 + y_2^2 + \overline{\lambda_1}\,(4\overline{x_1} + y_2 - 1)$$

$$\boldsymbol{s.t} \qquad \boldsymbol{\overline{x_1} + 4y_2 - 1 = 0} \tag{8.8}$$

In this example there only two groups (*a* = 1,*2*) of subsystems and additionally all constraints are equality ones.

The OCD algorithm 2 is applied below.

Step 0: Initialization.

Variables and multipliers are initialized in each block (*a* = 1,*2*), i.e,

$$x = \begin{pmatrix} 0.4 \\ 0.4 \end{pmatrix} \qquad y = \begin{pmatrix} 0.4 \\ 0.4 \end{pmatrix} \qquad \lambda = \begin{pmatrix} -0.01 \\ -0.01 \end{pmatrix} \tag{8.9}$$

Step 1: Single iteration.

Subsystem 1: System X

Subsystem 1 carries out one single iteration. It computes a movement direction for the first decomposed subproblem, using Newton's method, for $x = \bar{x}$. The Lagrangian function for this problem is

$$\mathcal{L}_x(x_1, x_{2,}\lambda_1) = x_1^2 + x_2^2 + \overline{\lambda_2}\,(x_1 + 4\overline{y_2} - 1) + \lambda_1(x_1 + 4\overline{y_2} - 1)$$
$$\mathcal{L}_x(x_1, x_{2,}\lambda_1) = x_1^2 + x_2^2 - 0.01x_1 + \lambda_1(x_1 + 0.4 - 1) \tag{8.10}$$

Then,

$$\nabla_{x1,x2,\lambda1}\mathcal{L}_x(x_1, x_{2,}\lambda_1) = \begin{pmatrix} 2x_1 - 0.01 + 4\lambda_1 \\ 2x_2 \\ 4x_1 + 0.4 - 1 \end{pmatrix}$$

$$\nabla_{x1,x2,\lambda1}\mathcal{L}_x(0.4,0.4,-0.01) = \begin{pmatrix} 0.75 \\ 0.80 \\ 1.00 \end{pmatrix} \tag{8.11}$$

$$\nabla^2{}_{x1,x2,\lambda1}\mathcal{L}_x(x_1,x_{2,}\lambda_1)=\begin{pmatrix}2&0&4\\0&2&0\\4&0&0\end{pmatrix} \tag{8.12}$$

If the Newton's method is applied

$$\nabla^2{}_{x1,x2,\lambda1}\mathcal{L}_x\begin{pmatrix}\Delta x_1\\\Delta x_2\\\Delta\lambda_1\end{pmatrix}=-\nabla_{x1,x2,\lambda1}\mathcal{L}_x$$

$$\begin{pmatrix}2&0&4\\0&2&0\\4&0&0\end{pmatrix}\begin{pmatrix}\Delta x_1\\\Delta x_2\\\Delta\lambda_1\end{pmatrix}=-\begin{pmatrix}0.75\\0.80\\1.00\end{pmatrix} \tag{8.13}$$

where the result is

$$\begin{pmatrix}\Delta x_1\\\Delta x_2\\\Delta\lambda_1\end{pmatrix}=\begin{pmatrix}-0.25\\-0.40\\-0.0625\end{pmatrix} \tag{8.14}$$

Subsystem 2: System Y

Subsystem 2 carries out one single iteration. It computes a movement direction for the first decomposed subproblem, using Newton's method, for $y=\bar{y}$. The Lagrangian function for this problem is

$$\mathcal{L}_y(y_1,y_{2,}\lambda_2)=y_1^2+y_2^2+\overline{\lambda_1}\,(4\overline{x_1}+y_2-1)+\lambda_{\,2}(\overline{x_1}+4y_2-1)$$
$$\mathcal{L}_y(y_1,y_{2,}\lambda_2)=y_1^2+y_2^2-0.01y_2+\lambda_{\,2}(0.4+4y_2-1) \tag{8.15}$$

Then,

$$\nabla_{y1,y2,\lambda2}\mathcal{L}_y(y_1,y_{2,}\lambda_2)=\begin{pmatrix}2y_1\\2y_2-0.0796+4\lambda_{\,2}\\4y_2-0.8\end{pmatrix}$$

$$\nabla_{y1,y2,\lambda2}\mathcal{L}_y(0.4{,}0.4,-0.01)=\begin{pmatrix}0.80\\0.75\\1.00\end{pmatrix} \tag{8.16}$$

$$\nabla^2{}_{y1,y2,\lambda2}\mathcal{L}_y(y_1,y_{2,}\lambda_2)=\begin{pmatrix}2&0&4\\0&2&4\\0&4&0\end{pmatrix} \tag{8.17}$$

If the Newton's method is applied

$$\nabla^2{}_{y1,y2,\lambda2}\mathcal{L}_y\begin{pmatrix}\Delta y_1\\\Delta y_2\\\Delta\lambda_2\end{pmatrix}=-\nabla_{y1,y2,\lambda2}\mathcal{L}_y$$

$$\begin{pmatrix}2&0&4\\0&2&4\\0&4&0\end{pmatrix}\begin{pmatrix}\Delta y_1\\\Delta y_2\\\Delta\lambda_2\end{pmatrix}=-\begin{pmatrix}0.80\\0.75\\1.00\end{pmatrix} \tag{8.18}$$

where the result is

$$\begin{pmatrix}\Delta y_1\\ \Delta y_2\\ \Delta\lambda_2\end{pmatrix}=\begin{pmatrix}-0.40\\ -0.25\\ -0.0625\end{pmatrix} \tag{8.19}$$

Step 2: Updating.

Each subsystem (*a* = 1, 2) updates its variables and parameters

Subsystem 1: System X

$$x = x + \Delta x = \begin{pmatrix}0.4\\ 0.4\end{pmatrix} + \begin{pmatrix}-0.25\\ -0.40\end{pmatrix} = \begin{pmatrix}0.15\\ 0.00\end{pmatrix} \tag{8.20}$$

$$\lambda_1 = \lambda_1 + \Delta\lambda_1 = (-0.01) + (-0.0625) = (-0.0725) \tag{8.21}$$

Subsystem 2: System Y

$$y = y + \Delta y = \begin{pmatrix}0.4\\ 0.4\end{pmatrix} + \begin{pmatrix}-0.40\\ -0.25\end{pmatrix} = \begin{pmatrix}0.00\\ 0.15\end{pmatrix} \tag{8.22}$$

$$\lambda_2 = \lambda_2 + \Delta\lambda_2 = (-0.01) + (-0.0625) = (-0.0725) \tag{8.23}$$

Step 3: Convergence.

The algorithm stops if variables do not change significantly in two consecutive iterations.
In this exercise we assume that the selected convergence condition $\|h(x,y)\| < 10^{-4}$ is satisfied (h(x,y) from Eq.)

$$h(x1,x2,y1,y2) = h(0.15,0,0,0.15) = \begin{pmatrix}-0.25\\ -0.25\end{pmatrix}$$
$$\|h\| = 0.3536 > 10^{-4} \tag{8.24}$$

As the convergence condition is not satisfied, variables and multipliers are fixed and Steps 1, 2, and 3 of the algorithm are repeated until convergence is achieved,

$$\bar{x} = x = \begin{pmatrix}0.15\\ 0.00\end{pmatrix} \qquad \bar{y} = y = \begin{pmatrix}0.00\\ 0.15\end{pmatrix} \qquad \bar{\lambda} = \lambda = \begin{pmatrix}-0.0725\\ -0.0725\end{pmatrix} \tag{8.25}$$

Finally, the algorithm stops for k=7, with tolerance $\|h\| = 8.6317\ 10^{-5}$. The solution is

$$x^* = \begin{pmatrix}0.2\\ 0.0\end{pmatrix} \quad y^* = \begin{pmatrix}0.0\\ 0.2\end{pmatrix} \quad \lambda^* = \begin{pmatrix}-0.08\\ -0.08\end{pmatrix}$$

At the following table 8.2, the obtained results are presented:

Iteration	$f(x,y)$	x_1	x_2	y_1	y_2	λ_1	λ_2
1	0.045	0.150	0.000	0.000	0.150	-0.010	-0.010
2	0.090	0.212	0.000	0.000	0.212	-0.072	-0.072
3	0.077	0.197	0.000	0.000	0.197	-0.088	-0.088
4	0.081	0.201	0.000	0.000	0.201	-0.076	-0.076
5	0.079	0.200	0.000	0.000	0.200	-0.081	-0.081
6	0.080	0.200	0.000	0.000	0.200	-0.079	-0.079
7	0.080	0.200	0.000	0.000	0.200	-0.080	-0.080

Table 8.2 Evolution of the optimality condition decomposition (OCD) algorithm.

8.3.3 Sensitivity-based Coordination solution

Following the notation presented in § 7.3, the overall system optimal control problem is written as,

$$\Im(x_1, x_{2,} y_1, y_2) = minimize_{x_1, x_{2,} y_1, y_2}\ \phi(x_1, x_{2,} y_1, y_2) = \sum_{i=1}^{N} \phi_i(x_i, y_i)$$

$$s.t \quad \phi_1(x_1, x_2) = {x_1}^2 + {x_2}^2$$
$$\phi_2(y_1, y_2) = {y_1}^2 + {y_2}^2$$
$$c_1(x_1, x_{2,} y_1, y_2) = 4x_1 + y_2 - 1 = 0$$
$$c_2(x_1, x_{2,} y_1, y_2) = x_1 + 4y_2 - 1 = 0 \tag{8.26}$$

Then, after a decomposition process, the resulting subproblems to be solved could be summarized in two groups (*i* = 1,*2*):

$$\Im_1(x_1, x_{2,}) = minimize_{x1,x2} \quad x_1^2 + x_2^2$$

$$s.t \quad 4x_1 + y_2 - 1 = 0 \tag{8.27}$$

$$\Im_2(y_1, y_2) = minimize_{y1.y2}\ {y_1}^2 + {y_2}^2$$

$$\boldsymbol{s.t} \quad x_1 + 4y_2 - 1 = 0 \tag{8.28}$$

With subproblems (8.27-8.28), the sensitivity-based coordination algorithm is applied below.

Step 0: Initialization.

Set k=0. Feasible parameters $z_i{}^{[0]}$ and an initial guess of the Lagrange parameters $\lambda^{[0]}$ are chosen.

$$z_1{}^{[0]} = \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = \begin{pmatrix} 0.4 \\ 0.4 \end{pmatrix} \quad z_2{}^{[0]} = \begin{pmatrix} y_1 \\ y_2 \end{pmatrix} = \begin{pmatrix} 0.4 \\ 0.4 \end{pmatrix} \quad \lambda = \begin{pmatrix} \lambda_1 \\ \lambda_2 \end{pmatrix} = \begin{pmatrix} -0.01 \\ -0.01 \end{pmatrix} \tag{8.29}$$

Step 1: Parameters are communicated to all local controllers

Step 2: Solve the local optimization problems

$$f(z_i) = minimize_z\ \phi_i{}^*(z)$$
$$s.t \quad c_i(z) \geq 0$$

$$with\ the\ strictly\ convex\ objective\ function$$

$$\phi_i{}^*(z) = \phi_i(z) + \left[\sum_{\substack{j=1 \\ j\neq i}}^{N} \frac{\partial \phi_j}{\partial z_i}\Big\|_{z^{[k]}}^{T} - \lambda_j^{[k]T} \frac{\partial c_j}{\partial z_i}\Big\|_{z^{[k]}}\right]\left(z_i - z_i^{[k]}\right) \tag{8.30}$$

Subsystem 1: System Z1

$$f(z_1) = f(x_1, x_2) = minimize_{z_1}\ \phi_1{}^*(x_1, x_2)$$
$$s.t \quad 4x_1 + y_2 - 1 = 0$$

$$with\ the\ strictly\ convex\ objective\ function$$

$$\phi_1{}^*(x_1, x_2) = \phi_1(x_1, x_2) + \left[\begin{pmatrix} \frac{\partial \phi_2}{\partial x_1} \\ \frac{\partial \phi_2}{\partial x_2} \end{pmatrix}^T_{x_1^{[k]}, x_2^{[k]}} - \lambda_2^{[k]T} \begin{pmatrix} \frac{\partial c_2}{\partial x_1} \\ \frac{\partial c_2}{\partial x_2} \end{pmatrix}^T_{x_1^{[k]}, x_2^{[k]}}\right] \begin{pmatrix} x_1 - x_1^{[k]} \\ x_2 - x_2^{[k]} \end{pmatrix} \tag{8.31}$$

Subsystem 1 carries out one single iteration. We use Newton's method to solve this subsystem.The Lagrangian function for this problem is

$$\mathcal{L}_{z1}(x_1, x_{2,}\lambda_1) = \phi_1{}^*(x_1, x_2) + \lambda_1(4x_1 + y_2 - 1) \tag{8.32}$$

Then,

$$\nabla \mathcal{L}_{z1}(x_1, x_{2,}\lambda_1) = \begin{pmatrix} 2x_1 + 0.1330 + 4\lambda_1 \\ 2x_2 \\ 4x_1 + y2 - 1 \end{pmatrix} \tag{8.33}$$

$$\nabla_{x1,x2,\lambda1} \mathcal{L}_{z1}(0.4, 0.4, -0.01) = \begin{pmatrix} 0.77 \\ 0.80 \\ 1.00 \end{pmatrix} \tag{8.34}$$

ETSEIB

$$\nabla^2{}_{x1,x2,\lambda1}\mathcal{L}_{z1}(x_1,x_{2,}\lambda_1)=\begin{pmatrix}2&0&4\\0&2&0\\4&0&0\end{pmatrix} \tag{8.35}$$

If the Newton's method is applied

$$\nabla^2{}_{x1,x2,\lambda1}\mathcal{L}_{z1}\begin{pmatrix}\Delta x_1\\\Delta x_2\\\Delta\lambda_1\end{pmatrix}=-\nabla\mathcal{L}_{z1}$$

$$\begin{pmatrix}2&0&4\\0&2&0\\4&0&0\end{pmatrix}\begin{pmatrix}\Delta x_1\\\Delta x_2\\\Delta\lambda_1\end{pmatrix}=-\begin{pmatrix}0.77\\0.80\\1.00\end{pmatrix} \tag{8.36}$$

where the result is

$$\begin{pmatrix}\Delta x_1\\\Delta x_2\\\Delta\lambda_1\end{pmatrix}=\begin{pmatrix}-0.25\\-0.40\\-0.0675\end{pmatrix} \tag{8.37}$$

Subsystem 2: System Z2

$$f(z_2)=f(y_1,y_2)=minimize_{z_2}\ \phi_2{}^*(y_1,y_2)$$
$$s.t\quad x_1+4y_2-1=0$$

$$with\ the\ strictly\ convex\ objective\ function$$

$$\phi_2{}^*(y_1,y_2)=\phi_1(y_1,y_2)+\left[\begin{pmatrix}\frac{\partial\phi_1}{\partial y_1}\\\frac{\partial\phi_2}{\partial y_2}\end{pmatrix}^T_{y_1^{[k]},y_2^{[k]}}-\lambda_1^{[k]T}\begin{pmatrix}\frac{\partial c_1}{\partial y_1}\\\frac{\partial c_1}{\partial y_2}\end{pmatrix}^T_{y_1^{[k]},y_2^{[k]}}\right]\begin{pmatrix}y_1-y_1^{[k]}\\y_2-y_2^{[k]}\end{pmatrix} \tag{8.38}$$

Subsystem 1 carries out one single iteration. We use Newton's method to solve this subsystem.The Lagrangian function for this problem is

$$\mathcal{L}_{z2}(y_1,y_{2,}\lambda_1)=\phi_2{}^*(y_1,y_2)+\lambda_2(x_1+4y_2-1) \tag{8.39}$$

Then,

$$\nabla\mathcal{L}_{z2}(y_1,y_{2,}\lambda_1)=\begin{pmatrix}2y_1\\2y_2+0.01+4\lambda_2\\x_1+4y2-1\end{pmatrix} \tag{8.40}$$

$$\nabla_{y1,y2,\lambda2}\mathcal{L}_{z2}(0.4,0.4,-0.01)=\begin{pmatrix}0.80\\0.77\\0.75\end{pmatrix} \tag{8.41}$$

$$\nabla^2{}_{y1,y2,\lambda2}\mathcal{L}_{z2}(y_1,y_{2,}\lambda_2)=\begin{pmatrix}2&0&0\\0&2&4\\0&4&0\end{pmatrix} \tag{8.42}$$

If the Newton's method is applied

$$\nabla^2{}_{y1,y2,\lambda 2}\mathcal{L}_{z2}\begin{pmatrix}\Delta y_1\\ \Delta y_2\\ \Delta\lambda_2\end{pmatrix} = -\nabla\mathcal{L}_{z2}$$

$$\begin{pmatrix}2 & 0 & 0\\ 0 & 2 & 4\\ 0 & 4 & 0\end{pmatrix}\begin{pmatrix}\Delta y_1\\ \Delta y_2\\ \Delta\lambda_2\end{pmatrix} = -\begin{pmatrix}0.80\\ 0.77\\ 0.75\end{pmatrix} \tag{8.43}$$

where the result is

$$\begin{pmatrix}\Delta y_1\\ \Delta y_2\\ \Delta\lambda_2\end{pmatrix} = \begin{pmatrix}-0.40\\ -0.1875\\ -0.0988\end{pmatrix} \tag{8.44}$$

Step 3: Set k = k + 1

Each subsystem (*i* = 1, 2) updates its variables and parameters

Subsystem 1: System X

$$z_1{}^{[1]} = \begin{pmatrix}x_1\\ x_2\end{pmatrix} = \begin{pmatrix}x_1\\ x_2\end{pmatrix} + \Delta x = \begin{pmatrix}0.4\\ 0.4\end{pmatrix} + \begin{pmatrix}-0.25\\ -0.40\end{pmatrix} = \begin{pmatrix}0.15\\ 0.00\end{pmatrix} \tag{8.45}$$

$$\lambda_1 = \lambda_1 + \Delta\lambda_1 = (-0.01) + (-0.0625) = (-0.0775) \tag{8.46}$$

Subsystem 2: System Y

$$z_2{}^{[1]} = \begin{pmatrix}y_1\\ y_2\end{pmatrix} + \Delta y = \begin{pmatrix}0.4\\ 0.4\end{pmatrix} + \begin{pmatrix}-0.40\\ -0.25\end{pmatrix} = \begin{pmatrix}0.00\\ 0.2125\end{pmatrix} \tag{8.47}$$

$$\lambda_2 = \lambda_2 + \Delta\lambda_2 = (-0.01) + (-0.0625) = (-0.1088) \tag{8.48}$$

Step 4: Stop.

The algorithm stops if variables do not change significantly in two consecutive iterations.

In this exercise we assume that the selected convergence condition

$\epsilon_{rel} = \frac{\| z^{[k]} - z^{[k-1]} \|_2}{\| z^{[k]} \|_2} < 10^{-4}$ is satisfied.

$$\epsilon_{rel} = 2.4846 > 10^{-4} \tag{8.49}$$

As the convergence condition is not satisfied, variables and multipliers are fixed and Steps 0,1, 2, and 3 of the algorithm are repeated until convergence is achieved,

Finally, the algorithm stops for k=5, with tolerance $\|\epsilon_{rel}\| = 4.1706\ 10^{-5}$. The solution is

$$x^* = \begin{pmatrix}0.2\\ 0.0\end{pmatrix} \quad y^* = \begin{pmatrix}0.0\\ 0.2\end{pmatrix} \quad \lambda^* = \begin{pmatrix}-0.1332\\ -0.1332\end{pmatrix} \tag{8.50}$$

At the following table 6.1 obtained results are presented:

Iteration	$f(x,y)$	x_1	x_2	y_1	y_2	λ_1	λ_2
1	0.068	0.150	0.000	0.000	0.213	-0.077	-0.109
2	0.079	0.197	0.000	0.000	0.201	-0.125	-0.127
3	0.079	0.199	0.000	0.000	0.200	-0.132	-0.132
4	0.080	0.200	0.000	0.000	0.200	-0.133	-0.133
5	0.080	0.200	0.000	0.000	0.200	-0.133	-0.133

Table 8.3 Evolution of the sensitivity-based coordination decomposition algorithm.

8.4 Centralised MPC Simulation results

In this section centralised simulation results will be exposed and commented for both building description.

First the operating point is defined as:

- Zone temperature : $T_i^{(0)} = 23℃ (i = 1, \dots, 4)$
- Supply mass airflows: $\dot{m}_i^{(0)} = 0.192 (i = 1, \dots, 4)$
- Supply air-temperature : $T_s^{(0)} = 26℃ (i = 1, \dots, 4)$
- Initial outside air temperature: $T_{oa}^{(0)} = 5℃$
- Outside temperature variation is represented in Fig. (8.5)
- Internal heat gain, accumulative heat flux due to occupants and electronic devices, are shown in Fig. (8.5)

As explained in § 6.3.1, the continuous-time Non-Linear state space model is linearized around the above operational point. This linearized model is discretized with a sampling period h=60s.

Thermal comfort range is stipulated in $23℃ \pm 0.5$ and supply airflow to each zone varies between 0.0192 kg/s and 0.31 kg/s.

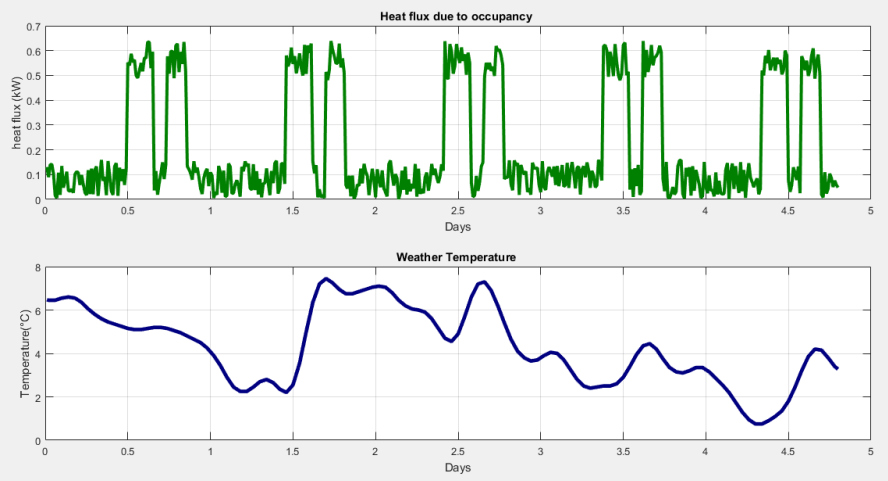


Fig. 8.5 Heat flux due to occupancy and outside temperature

Simulation results are provided for VAV-type building configuration, in which CMPC controls the thermal comfort in the building. The test is performed during 5 days and model predictive controller solves the energy optimization problem over a prediction horizon of 24 hours. Also related energy cost per kWh are presented:

- Related energy cost per kWh required at the heating coil to deliver an airflow at temperature T_s is $c_1 = 0.5$
- Related energy cost per kWh for central supply fan at AHU is $c_2 = 1$

8.4.1 Building of 8 zones CMPC

Temperature responses T_i of the eight zones building

There is a clearly defined pattern to the following figure 8.6, and this can be taken to mean that the supply air-temperature is constantly maintained within the desired comfort band. So a tendency to remain steady is observed in all zones with any significant change thanks to a well-adjusted control strategy. As it can be seen from figure 8.7, this tendency is confirmed in zone 1 for example with more detail. Temperature is set in $23℃$ and there is no tentative to exceed thermal comfort limits.

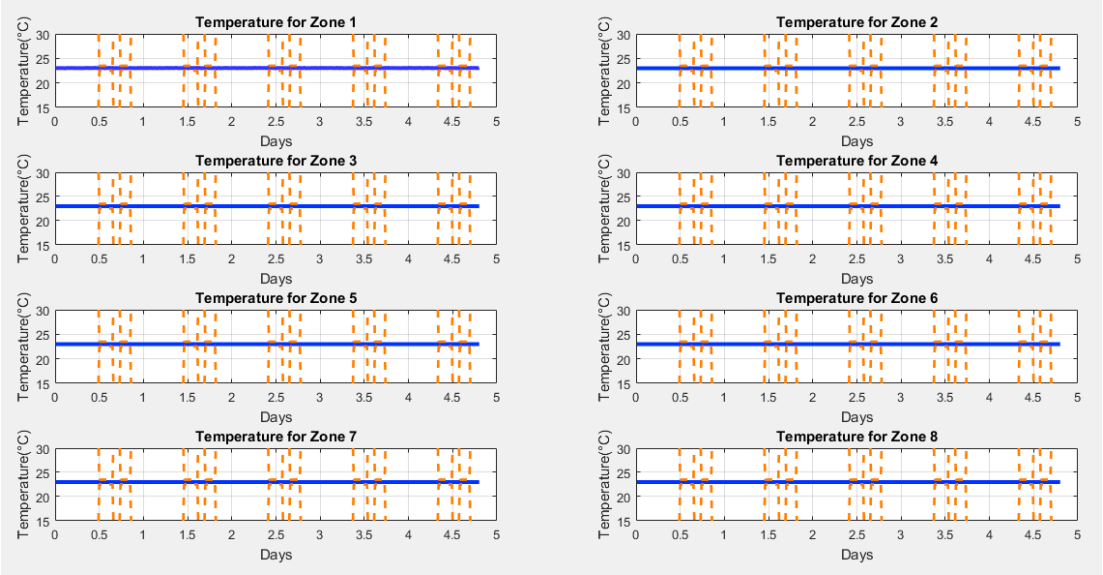


Fig. 8.6 Temperature response for all 8 zones

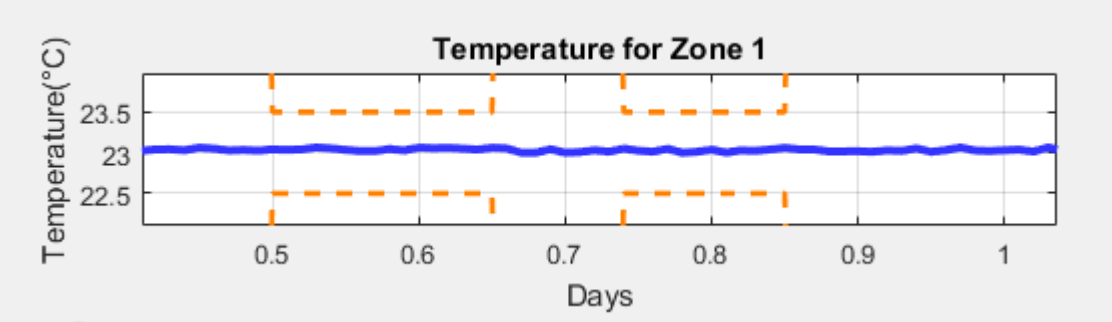


Fig. 8.7 Temperature for zone 1 without exceeding comfort range

Computed optimal set points for the supply airflows of the eight zones

It is worth observing that the supply air-temperature is modified in a way such that that the temperature in all zones are effectively maintained within the comfort band. For this reason, figure 8.8 shows a relevant comparative between the supply air flow rate sent and the internal heat gain in one zone. Undoubtedly, one can interpret how the computed supply airflows set points fall when a climbing tendency starts at the internal gains heat flux and vice versa. This well-suited airflow compensation strategy is observed during all middays and let us confirm that temperature of zone is maintained at comfort levels.

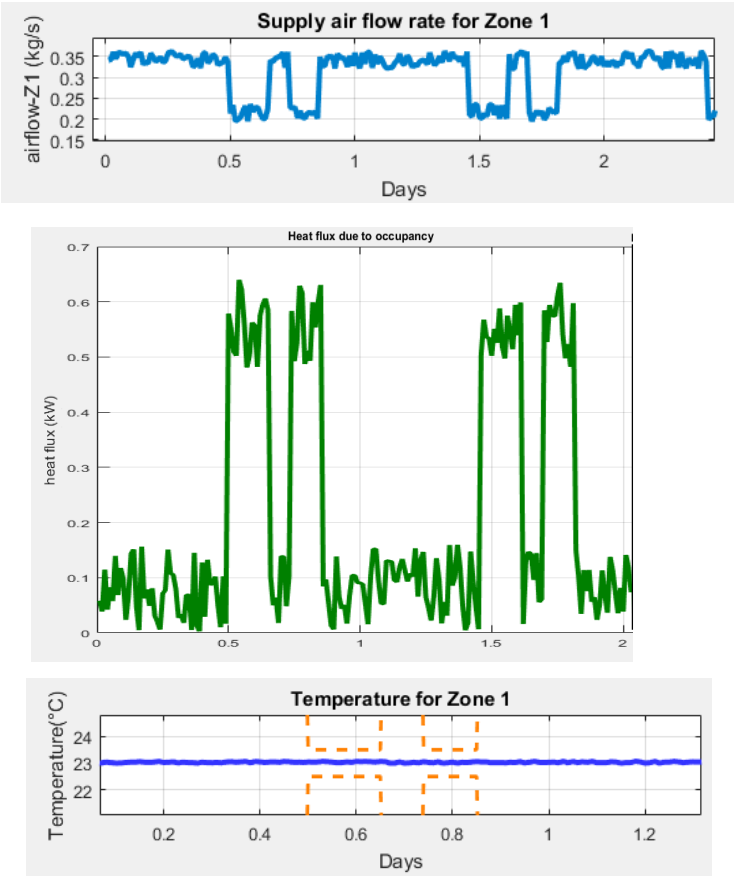


Fig. 8.8 Effect of occupancy on temperature and supply air flow in one zone

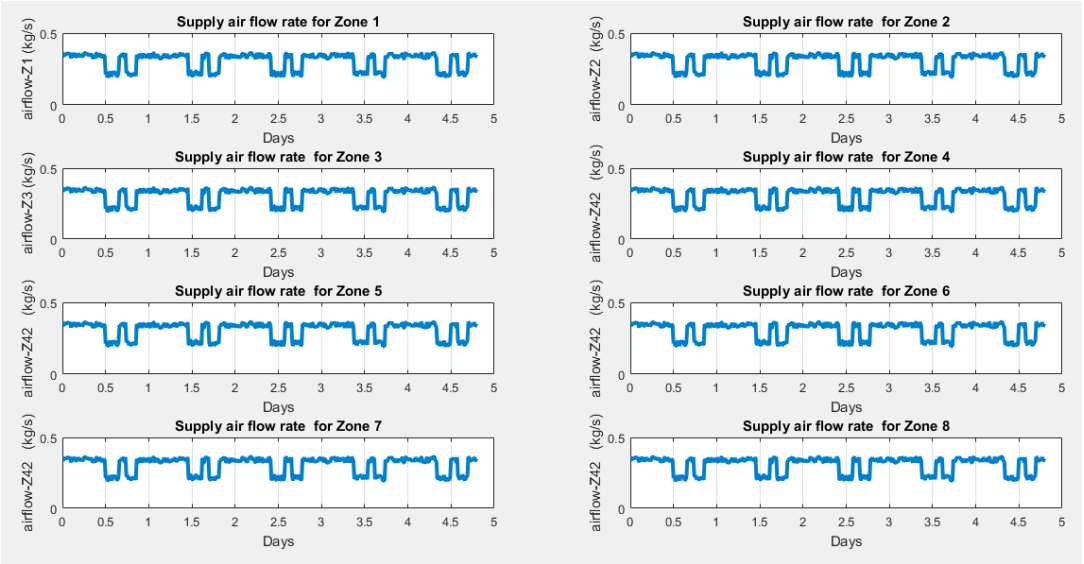


Fig. 8.9 Supply air flow set points in all 8 zones

8.4.2 Building of 6 zones CMPC

Simulation results for the VAV centralised 6 rooms office building are quite similar from the above case.

Temperature responses Ti of the six zones building

It can be seen that zone temperatures are maintained at the thermal comfort range during the presence of occupants. This tendency is observed in all 6 zones. Fig. 8.10, that represents zone 1, is set as an example for this explained behaviour.

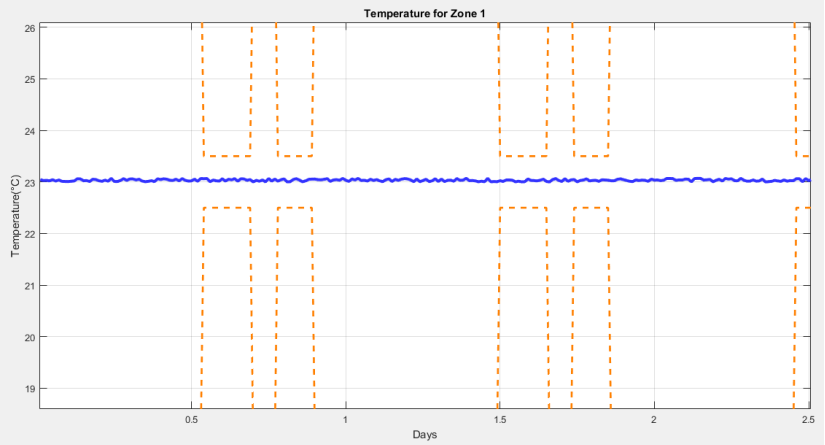


Fig. 8.10 Temperature for zone 1

Computed optimal set points for the supply airflows of the six zones

The supply air flow suffers modifications in order to compensate changes of heat flux due to occupancy. Fig. 8.11, that represents zone 1, is set as an example for this explained behaviour.

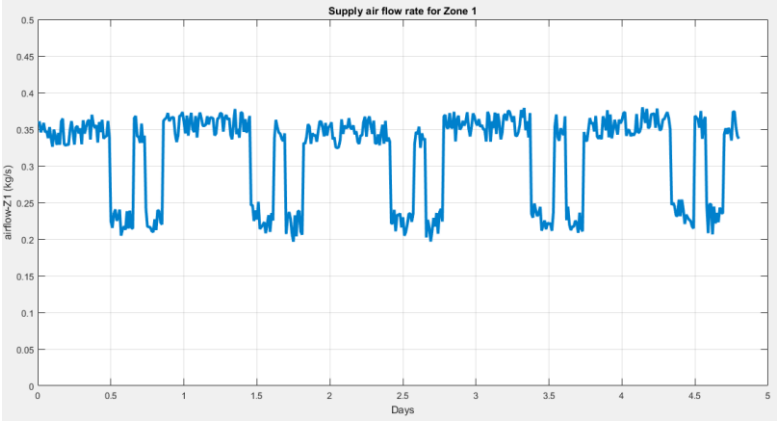


Fig. 8.11 Supply air flow rate in zone 1

8.5 Distributed MPC Simulation results

8.5.1 Optimal Conditions Decomposition (OCD) (Coupling of states x)

In this section simulation results will be exposed for the proposed distributed control approach on benchmark building of six zones. Applying OCD algorithm for the centralised problem, we evaluate KKT system matrix for initial values of $(x^{(0)}, u^{(0)})$. Partitioning the KKT matrix, we obtain the subproblems with the following two groups (p = 2) of zones as $\{1,2,3\}$ and $\{4,5,6\}$.

In this case the coupling between subsystems is made throw the state temperatures x. The simulation results for the achieved decomposition are explained as follows.

OCD Temperature responses Ti of the six zones building

In general, with the decentralized techniques we can observe that temperature is no longer constant. So there is in Figure 8.12 a sharp decrease in temperature during hours where there is not a strong comfort solicitation. Then, a substantial rise appears when the desired comfort band must be followed and temperature define nervous pattern trajectory. Temperature changes between $22 - 23℃$ and approaches dangerously thermal comfort limits. This tendency compared with the centralized solution is an important issue to improve.

As it can be seen from figure 8.13, this tendency confirmed in all six zones with more detail.

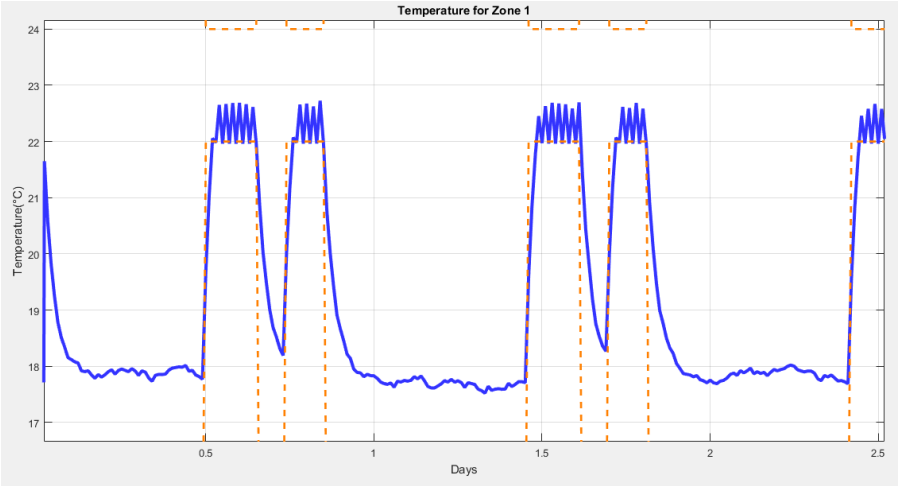


Fig. 8.12 Temperature for zone 1

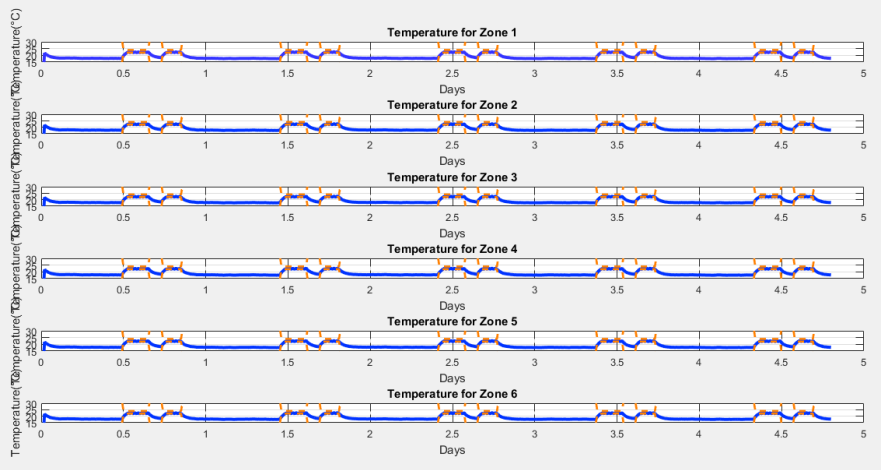


Fig. 8.13 Temperature response for all 6 zones

Computed optimal setpoints for the supply airflows u of the six zones

It is worth observing that the supply air-temperature is modified in a way to meet the irregular variation of temperature in all zones. We can observe in Figure 8.14 how supply air flow fall and drop constantly during the presence of occupants in the respectively zone.

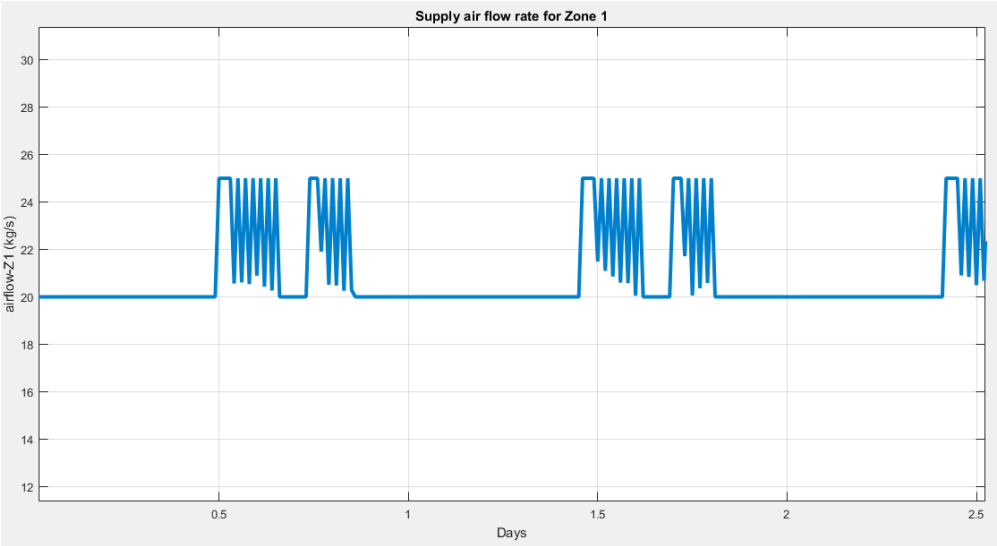


Fig. 8.14 Supply air flow rate in zone 1

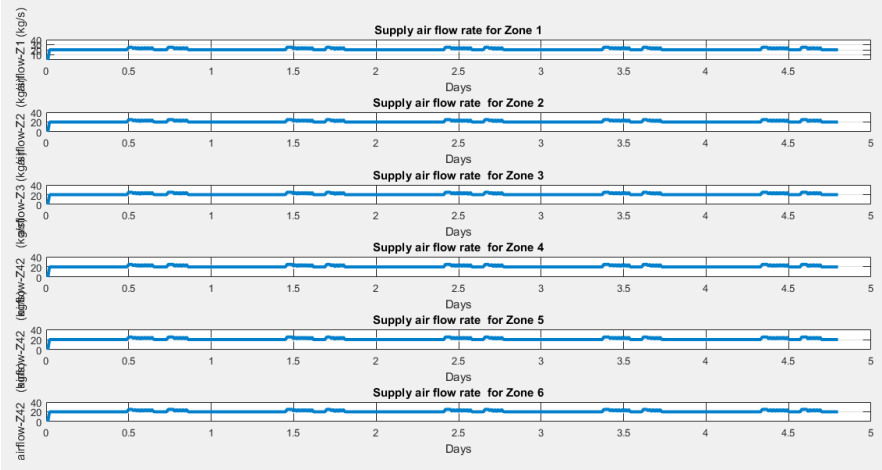


Fig. 8.15 Supply air flow set points in all 6 zones

8.5.2 Sensitivity-based coordination (Coupling of input u)

In this section simulation results will be exposed for the sensitivity- based coordination distributed control on the benchmark building of eight zones. Applying S-DMPC algorithm, explained in § 7.3.3, the overall optimality is archived by extending the local objective functions by linear approximations of the contributions of the neighbouring systems.

Like other approaches, problem is linearized around an operating point $(x_i^{(0)}, u_i^{(0)})$. Main problem is also decomposed in the following two groups (p = 2) of zones as $\{1,2,3\}$ and $\{4,5,6\}$.

However, in this case the coupling between subsystems is made throw the inputs u. The simulation results for the achieved decomposition are explained as follows.

S-DMPC Temperature responses Ti of the eight zones building

With this decomposition technique we can observe that overall temperature is maintained in the thermal comfort during the presence of occupants in the building. Nevertheless, it is important to remark that there is a slight decrease of temperature during hours where there is not a strong comfort solicitation helping to save energy. Also, temperature follow a steady rise at the comfort band. Figure (8.16) show the temperature in zone 1 as an example for the explained behaviour.

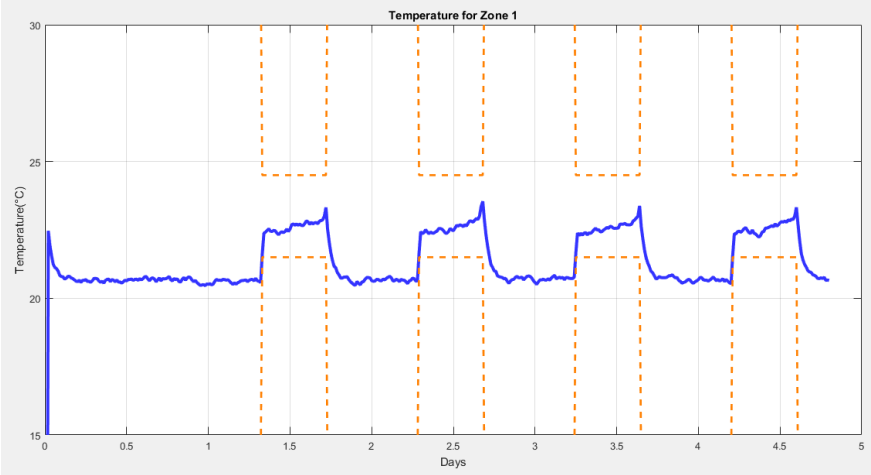


Fig. 8.16 Temperature for zone 1

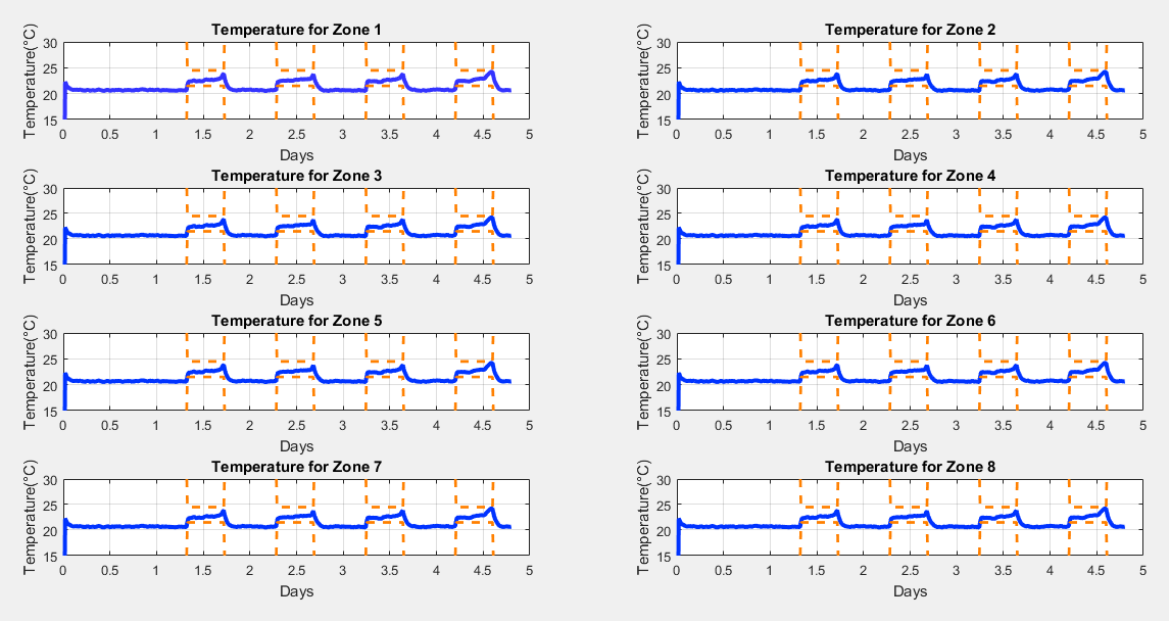


Fig. 8.17 Temperature response for all 8 zones

S-DMPC Computed optimal setpoints for the supply airflows of the eight zones

In this section key point values of the required supply airflow controlled by heating coil in FCU are shown. In fact, the supply air-temperature is modified in a way to meet variation requirements of temperature in all zones. In this case, supply flow fall sharply when there is more presence of occupants in order to compensate the internal gains heat flux.

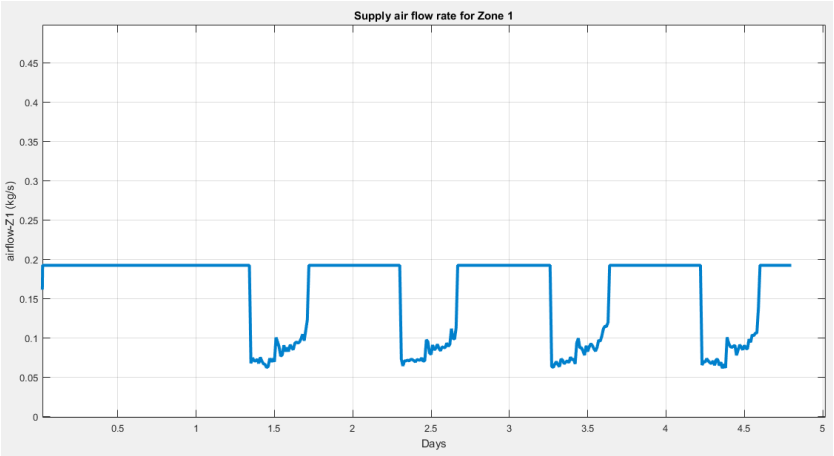


Fig. 8.18 Supply air flow rate in zone 1

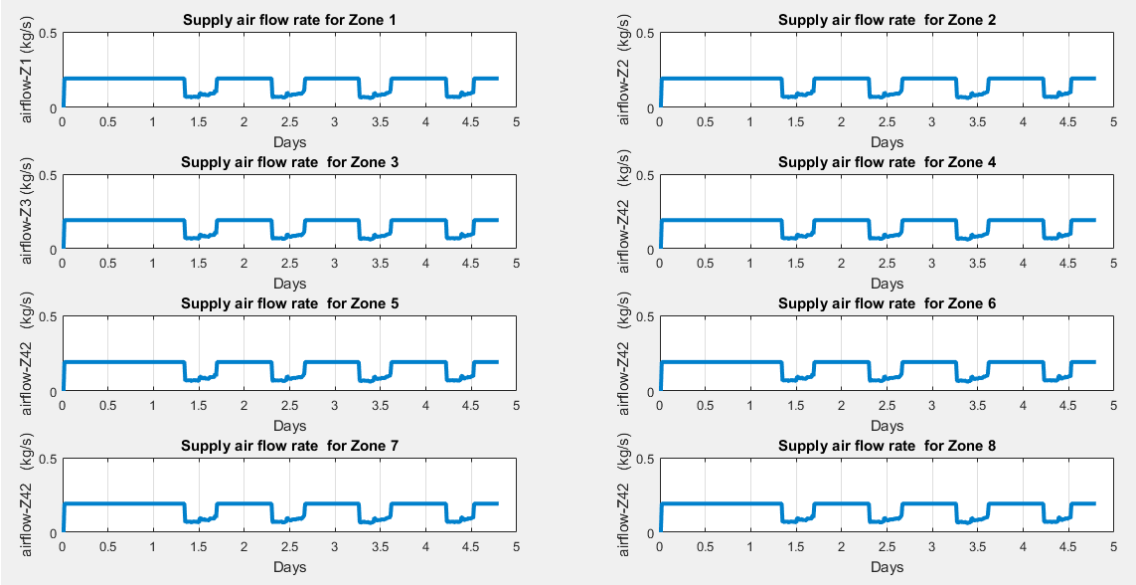


Fig. 8.19 Supply air flow set points in all 8 zones

Conclusions

The main objective of this project was to try to make a complete comparison for all explained techniques throughout the memory for a single type of building. As discussed in Chapter 8, we started the project with a considerable disadvantage as having different building results with different variable couplings and functions.

The key to overcome these difficulties was to maintain the structure of the MPC control loop, but it was necessary to prepare changes in the internal optimization models and problems of each controller in order to solve the same energy optimization problem. This is why the author, after understanding all decomposition techniques with illustrative examples, tried to modify the codes of simulation problems to achieve this goal.

Finally, it was not possible to achieve a single LSS Building problem with all non-centralized and centralized configuration because in fact these problems are not equivalent. Although the codes were successfully modified, the result were not concluding. (See Appendix for more details on Matlab-Yalmip MPC code formulation)

Future work should be to concrete in which cases it is possible to a stablish a close relation between two different energy optimization problems as to analyse coupling, type of separable or non-separable objective function and a proper analysis of linear and nonlinear problems.

Acknowledgement

I would like to express my gratitude and appreciation to all those who gave me the possibility to complete this report. A special thanks to my Final Degree Project director, Vicenç Puig, whose help, meetings, suggestions and encouragement helped me to coordinate my project.

Last but not least, many thanks go to my parents that have given me their full support and effort in guiding me throughout this Industrial Engineering Degree from the very first day. They have given me the courage, energy and determination to overcome many obstacles all through these years to be able to achieve my goals.

Bibliography

References

[1] HALVGAARD, Rasmus Fogtmann; JØRGENSEN, John Bagterp; POULSEN, Niels Kjølstad; MADSEN, Henrik . *Model Predictive Control for Smart Energy Systems*, Copenhagen: Technical University of Denmark- 2014 [Cons: April 2017]

[2] ELLIOT, Matthew Stuart. *Decentralized model predictive control of a multiple evaporator HVAC system*. Texas: A&M University- 2008 [Cons: May 2017]

[3] CARANTU, Constantin Florin. *Model-based Predictive Control Project.* University of Lasi 2014 [Cons: May 2017]

[4] MAYNE, RAWLINGS, RAO, SCOKAERT. *Constrained model predictive control : Stability and optimality.* Automatica 2000 [Cons: April 2017]

[5] COMMUNICATION FROM THE COMMISSION TO THE EUROPEAN PARLIAMENT, THE COUNCIL,THE EUROPEAN ECONOMIC AND SOCIAL COMITTE AND THE COMITTE OF THE REGIONS. Brussel 2010.

[6] ROWELL, Derek. *State-Space Representation of LTI Systems.* MIT Massachusetts Technical Institute October 2002 [Cons: May 2017]

[7] Simply VAV. *About VAV* [Webpage] [Cons: May 2017] [http://web.mit.edu/2.14/www/Handouts/StateSpace.pdf]

[8] Heinen & Hopman. *Merchant Fan Coil Unit* [Webpage] [Cons: May 2017] [https://heinenhopman.com/en/merchant/fan-coil-unit/]

[9] CONEJO, Antonio J. *Decomposition Techniques in Mathematical Programming* Springer 2006 [Cons: April 2017]

[10] SCHEU, Holger ;MARQUART Wolfgang. *Sensitivity-based coordination in distributed model predictive control.* Journal of Process Control volume-21 2011 [Cons: April 2017]

[11] DARURE, Tejaswinee ; YAMÉ, Joseph J.; HAMELIN Fréderic. Fault-Adaptive Control of VAV damper stuck in a Multizone Building. Nancy: Université de Lorraine. Centre de Recherche en Automatique de Nancy [Cons: March 2017]

Other consulted sources

[12] CIGLER,Jirí. *Model Predictive Control for Building*, Prague: Czech Technical University in Prague – June 2013 [Cons: April 2017]

[13] DONG,B.; LAM,K.P. *A real-time model predictive control for building heating and cooling systems based on the occupancy behaviour pattern detection and local weather forecasting*, 2014 [Article] [Cons: April 2017]

[14] VASAK,Mario; STARCIC,Antonio; MARTINCEVIC,Anita. *Model predictive control of heating in a family house*, 2014. Zagreb: University of Zagreb [Cons: April 2017]

[15] RODRÍGUEZ, Francisco; BERENGUEL, Manuel; GUZMAN Jose Luis; RAMÍREZ-ARIAS, Armando. *Modeling and Control of Greenhouse Crop Growth*" Springer 2014 [Cons: May 2017]

[16] KAVGIC, Miroslava; SWAN,Lukas. *Model predictive control for commercial buildings: trends and opportunities*" 2016 [Book] [Cons: May 2017]

[17] BARREIRO, Julián. *Optimization-based Control with Population Dynamics for Large Scale Complex Systems.* Barcelona: UPC Universitat Politècnica de Catalunya September 2014 [Doctoral Thesis] [Cons: May 2017]

[18] BARREIRO, Julián. *Optimization-based Control with Population Dynamics for Large Scale Complex Systems.* Barcelona: UPC Universitat Politècnica de Catalunya September 2014 [Doctoral Thesis] [Cons: May 2017]

[19] BARCELLI, Davide *Decentralized and Hierarchical Model Predictive Control of Networked Systems.* Siena: Università Degli Studi di Siena October 2012 [Doctoral Thesis] [Cons: May 2017]

[20] CAMACHO,E.F ; BORDONS, C. *Model Predictive Control.* Springer-Verlag 2007

Abstract

These are the appendices from the final degree project Centralized and Non-Centralized Model Predictive Control of a Multizone Building. This part describes the details Matlab-Yalmip codes that have been formulated in order to have centralized CMPC configuration, Optimal Conditions Decomposition and Sensitivity-Coordination DMPC at the same building configuration. In this case, the author has tried to work with the 8 zones building because the coupling of state variables in the other office building make much more difficult to work with.

SUMMARY

APPENDIX A: Codes of Illustrative Example

A.1 Centralized Illustrative example

```
% Inicialitzation
Z1 = 2;
Z2 = 3;
l1 = 5;
l2 = 6;
Z = [ Z1, Z2, l1, l2];
syms z1 z2 lambda1 lambda2
h1 = z1^2+z2^2-12.5572;
h2 = z1 + z2 - 5;
h = 0
f = - (20*z1 + 16*z2 - 2*(z1)^2 - (z2)^2)
% Lagrangian functions
L = -(20*z1 + 16*z2 - 2*(z1)^2 - (z2)^2)+ lambda2*(z1+z2-5)+
lambda1*(z1^2+z2^2-12.5572);
for k = 1 : 20
    Gradz1 = diff(L,z1);
    gz1=double(subs(Gradz1,{z1,z2,lambda1,lambda2},{Z1,Z2,l1,l2}));
    Gradz2 = diff(L,z2);
    gz2=double(subs(Gradz2,{z1,z2,lambda1,lambda2},{Z1,Z2,l1,l2}));
    Gradl1 = diff(L,lambda1);
    gl1=double(subs(Gradl1,{z1,z2,lambda1,lambda2},{Z1,Z2,l1,l2}));
    Gradl2 = diff(L,lambda2);
    gl2=double(subs(Gradl2,{z1,z2,lambda1,lambda2},{Z1,Z2,l1,l2}));
    G = [gz1 ;gz2 ; gl1 ;gl2];
    modG = sqrt ( gz1^2 + gz2^2 + gl1^2 + gl2^2)
    if modG < 10^(-4)
        'Problema has finished'
        break;
     end
    % Matriu hessiana
    Hz1z1 = diff(Gradz1,z1);
    hz1z1=double(subs(Hz1z1,{z1,z2,lambda1,lambda2},{Z1,Z2,l1,l2}));
    Hz1z2 = diff(Gradz1,z2);
    hz1z2=double(subs(Hz1z2,{z1,z2,lambda1,lambda2},{Z1,Z2,l1,l2}));
    Hz1l1 = diff(Gradz1,lambda1);
    hz1l1=double(subs(Hz1l1,{z1,z2,lambda1,lambda2},{Z1,Z2,l1,l2}));
    Hz1l2 = diff(Gradz1,lambda2);
    hz1l2=double(subs(Hz1l2,{z1,z2,lambda1,lambda2},{Z1,Z2,l1,l2}));
    H1 = [hz1z1 ,hz1z2 ,hz1l1 ,hz1l2];

    Hz2z1 = diff(Gradz2,z1);
    hz2z1=double(subs(Hz2z1,{z1,z2,lambda1,lambda2},{Z1,Z2,l1,l2}));
    Hz2z2 = diff(Gradz2,z2);
    hz2z2=double(subs(Hz2z2,{z1,z2,lambda1,lambda2},{Z1,Z2,l1,l2}));
    Hz2l1 = diff(Gradz2,lambda1);
    hz2l1=double(subs(Hz2l1,{z1,z2,lambda1,lambda2},{Z1,Z2,l1,l2}));
    Hz2l2 = diff(Gradz2,lambda2);
    hz2l2=double(subs(Hz2l2,{z1,z2,lambda1,lambda2},{Z1,Z2,l1,l2}));
    H2 = [hz2z1 ,hz2z2 ,hz2l1 ,hz2l2];

    Hl1z1 = diff(Gradl1,z1);
    hl1z1=double(subs(Hl1z1,{z1,z2,lambda1,lambda2},{Z1,Z2,l1,l2}));
```

```
    Hl1z2 = diff(Gradl1,z2);
    hl1z2=double(subs(Hl1z2,{z1,z2,lambda1,lambda2},{Z1,Z2,l1,l2}));
    Hl1l1 = diff(Gradl1,lambda1);
    hl1l1=double(subs(Hl1l1,{z1,z2,lambda1,lambda2},{Z1,Z2,l1,l2}));
    Hl1l2 = diff(Gradl1,lambda2);
    hl1l2=double(subs(Hl1l2,{z1,z2,lambda1,lambda2},{Z1,Z2,l1,l2}));
    H3 = [hl1z1 ,hl1z2 ,hl1l1 ,hl1l2];

    Hl2z1 = diff(Gradl2,z1);
    hl2z1=double(subs(Hl2z1,{z1,z2,lambda1,lambda2},{Z1,Z2,l1,l2}));
    Hl2z2 = diff(Gradl2,z2);
    hl2z2=double(subs(Hl2z2,{z1,z2,lambda1,lambda2},{Z1,Z2,l1,l2}));
    Hl2l1 = diff(Gradl2,lambda1);
    hl2l1=double(subs(Hl2l1,{z1,z2,lambda1,lambda2},{Z1,Z2,l1,l2}));
    Hl2l2 = diff(Gradl2,lambda2);
    hl2l2=double(subs(Hl2l2,{z1,z2,lambda1,lambda2},{Z1,Z2,l1,l2}));
    H4 = [hl2z1 ,hl2z2 ,hl2l1 ,hl2l2];
    H =  [H1 ;H2 ;H3 ;H4];

    % Metode Newton
    h = -inv(H) * G ;
    Z = Z + h';
    Z1=Z(1,1);
    Z2=Z(1,2);
    l1=Z(1,3);
    l2=Z(1,4);
end
Z
f=double(subs(f,{z1,z2},{Z1,Z2}))
```

A.2 Optimal Conditions Decomposition Illustrative example

```
% Inicialization
x1 = 0.4;
x2 = 0.4;
y1 = 0.4;
y2 = 0.4;
l1 = -0.01;
l2 = -0.01;
XP1 = [x1,x2,l1];
XP2 = [y1,y2,l2];
syms X1 X2 L1
LG1 = X1^2 + X2^2 + l2*(X1 + 4*y2 -1)+ L1*(4*X1 + y2 -1);
syms Y1 Y2 L2
LG2 = Y1^2 + Y2^2 + l1*(4*x1 + Y2 -1)+ L2*(x1 + 4*Y2 -1);
for k = 1 : 10
    LG1 = X1^2 + X2^2 + l2*(X1 + 4*y2 -1)+ L1*(4*X1 + y2 -1);
    LG2 = Y1^2 + Y2^2 + l1*(4*x1 + Y2 -1)+ L2*(x1 + 4*Y2 -1);
    % Gradient System1
    DLG1x1 = diff(LG1,X1);
    dlg1x1=double(subs(DLG1x1,{X1,X2,L1},{x1,x2,l1}));
    DLG1x2 = diff(LG1,X2);
    dlg1x2=double(subs(DLG1x2,{X1,X2,L1},{x1,x2,l1}));
    DLG1l1 = diff(LG1,L1);
    dlg1l1=double(subs(DLG1l1,{X1,X2,L1},{x1,x2,l1}));
    G1 = [dlg1x1;dlg1x2;dlg1l1];
```

```
% Hessiana System1
H1x1 = diff(DLG1x1,X1);
h1x1=double(subs(H1x1,{X1,X2,L1},{x1,x2,l1}));
H1x2 = diff(DLG1x1,X2);
h1x2=double(subs(H1x2,{X1,X2,L1},{x1,x2,l1}));
H1l1 = diff(DLG1x1,L1);
h1l1=double(subs(H1l1,{X1,X2,L1},{x1,x2,l1}));
H11 = [h1x1,h1x2,h1l1];

H2x1 = diff(DLG1x2,X1);
h2x1=double(subs(H2x1,{X1,X2,L1},{x1,x2,l1}));
H2x2 = diff(DLG1x2,X2);
h2x2=double(subs(H2x2,{X1,X2,L1},{x1,x2,l1}));
H2l1 = diff(DLG1x2,L1);
h2l1=double(subs(H2l1,{X1,X2,L1},{x1,x2,l1}));
H12 = [h2x1,h2x2,h2l1];

H3x1 = diff(DLG1l1,X1);
h3x1=double(subs(H3x1,{X1,X2,L1},{x1,x2,l1}));
H3x2 = diff(DLG1l1,X2);
h3x2=double(subs(H3x2,{X1,X2,L1},{x1,x2,l1}));
H3l1 = diff(DLG1l1,L1);
h3l1=double(subs(H3l1,{X1,X2,L1},{x1,x2,l1}));
H13 = [h3x1,h3x2,h3l1];
H1 = [H11;H12;H13];

% Newton's Method System 1
h1 = -inv(H1) * G1 ;
XP1 = XP1 + h1'
x1=XP1(1,1);
x2=XP1(1,2);
l1=XP1(1,3);

% Gradient System2
DLG2y1 = diff(LG2,Y1)
dlg2y1=double(subs(DLG2y1,{Y1,Y2,L2},{y1,y2,l2}));

DLG2y2 = diff(LG2,Y2)
dlg2y2=double(subs(DLG2y2,{Y1,Y2,L2},{y1,y2,l2}));

DLG2l2 = diff(LG2,L2)
dlg2l2=double(subs(DLG2l2,{Y1,Y2,L2},{y1,y2,l2}));
G2 = [dlg2y1;dlg2y2;dlg2l2];

% Hessiana System2
H1y1 = diff(DLG2y1,Y1);
h1y1=double(subs(H1y1,{Y1,Y2,L2},{y1,y2,l2}));
H1y2 = diff(DLG2y1,Y2);
h1y2=double(subs(H1y2,{Y1,Y2,L2},{y1,y2,l2}));
H1l2 = diff(DLG2y1,L2);
h1l2=double(subs(H1l2,{Y1,Y2,L2},{y1,y2,l2}));
H21 = [h1y1,h1y2,h1l2];

H2y1 = diff(DLG2y2,Y1);
h2y1=double(subs(H2y1,{Y1,Y2,L2},{y1,y2,l2}));
H2y2 = diff(DLG2y2,Y2);
```

```
    h2y2=double(subs(H2y2,{Y1,Y2,L2},{y1,y2,l2}));
    H2l2 = diff(DLG2y2,L2);
    h2l2=double(subs(H2l2,{Y1,Y2,L2},{y1,y2,l2}));
    H22 = [h2y1,h2y2,h2l2];

    H3y1 = diff(DLG2l2,Y1);
    h3y1=double(subs(H3y1,{Y1,Y2,L2},{y1,y2,l2}));
    H3y2 = diff(DLG2l2,Y2);
    h3y2=double(subs(H3y2,{Y1,Y2,L2},{y1,y2,l2}));
    H3l2 = diff(DLG2l2,L2);
    h3l2=double(subs(H3l2,{Y1,Y2,L2},{y1,y2,l2}));
    H23 = [h3y1,h3y2,h3l2];
    H2 = [H21;H22;H23];

    % Newton's Method System 2
    h2 = -inv(H2) * G2 ;
    XP2 = XP2 + h2'
    y1=XP2(1,1);
    y2=XP2(1,2);
    l2=XP2(1,3);
    % Convergence
    hP1 = 4*x1 + y2 - 1;
    hP2 = x1 + 4*y2 - 1;
    conver = sqrt ( hP1^2 + hP2^2);
    if conver < 10^(-4)
        'Problem has finished'
        break;
     end
end
```

A.3 Sensitivity-based coordination on Illustrative example

```
% Inicialization
x1 = 0.4;
x2 = 0.4;
y1 = 0.4;
y2 = 0.4;
l1 = -0.01;
l2 = -0.01;
X = [x1,x2,l1];
Y = [y1,y2,l2];
for k = 1 : 10
    x1old = x1;
    x2old = x2;
    y1old = y1;
    y2old = y2;
    syms X1 X2 Y1 Y2
    f1 = X1^2 + X2^2;
    c1 = 4*X1 + Y2 -1;
    f2 = Y1^2 + Y2^2;
    c2 = X1 + 4*Y2 -1;
    %% System 1 %%
    m11x1 = diff(f2,X1);
    m11x2 = diff(f2,X2);
    m11 = [m11x1,m11x2];
    m12x1 = diff(c2,X1);
    m12x2 = diff(c2,X2);
```

```
m12 = -l2*[m12x1,m12x2];
m1 = m11 + m12;
m2 = [X1-x1 , X2-x2];
m = m1*m2';
F1 = f1 + m;
syms L1
LG1 = F1 + L1*(c1);
% Gradient System1
DLG1x1 = diff(LG1,X1);
dlg1x1=double(subs(DLG1x1,{X1,X2,L1},{x1,x2,l1}));

DLG1x2 = diff(LG1,X2);
dlg1x2=double(subs(DLG1x2,{X1,X2,L1},{x1,x2,l1}));

DLG1l1 = diff(LG1,L1);
dlg1l1=double(subs(DLG1l1,{X1,X2,L1,Y1,Y2},{x1,x2,l1,y1,y2}));
G1 = [dlg1x1;dlg1x2;dlg1l1];

% Hessiana System1
H1x1 = diff(DLG1x1,X1);
h1x1=double(subs(H1x1,{X1,X2,L1},{x1,x2,l1}));
H1x2 = diff(DLG1x1,X2);
h1x2=double(subs(H1x2,{X1,X2,L1},{x1,x2,l1}));
H1l1 = diff(DLG1x1,L1);
h1l1=double(subs(H1l1,{X1,X2,L1},{x1,x2,l1}));
H11 = [h1x1,h1x2,h1l1];

H2x1 = diff(DLG1x2,X1);
h2x1=double(subs(H2x1,{X1,X2,L1},{x1,x2,l1}));
H2x2 = diff(DLG1x2,X2);
h2x2=double(subs(H2x2,{X1,X2,L1},{x1,x2,l1}));
H2l1 = diff(DLG1x2,L1);
h2l1=double(subs(H2l1,{X1,X2,L1},{x1,x2,l1}));
H12 = [h2x1,h2x2,h2l1];

H3x1 = diff(DLG1l1,X1);
h3x1=double(subs(H3x1,{X1,X2,L1},{x1,x2,l1}));
H3x2 = diff(DLG1l1,X2);
h3x2=double(subs(H3x2,{X1,X2,L1},{x1,x2,l1}));
H3l1 = diff(DLG1l1,L1);
h3l1=double(subs(H3l1,{X1,X2,L1},{x1,x2,l1}));
H13 = [h3x1,h3x2,h3l1];
H1 = [H11;H12;H13];

% Newton's Method System 1
h1 = -inv(H1) * G1;
X = X + h1'
x1=X(1,1);
x2=X(1,2);
l1=X(1,3);
```

```
%% System 2 %%
    n11y1 = diff(f1,Y1);
    n11y2 = diff(f1,Y2);
    n11 = [n11y1,n11y2];
    n12y1 = diff(c1,Y1);
    n12y2 = diff(c1,Y2);
    n12 = -l2*[n12y1,n12y2];
    n1 = n11 + n12;
    n2 = [Y1-y1 , Y2-y2];
    n = n1*n2';
    F2 = f2 + n;
    syms L2
    LG2 = F2 + L2*(c2);
    % Gradient System2
    DLG2y1 = diff(LG2,Y1);
    dlg2y1=double(subs(DLG2y1,{Y1,Y2,L2},{y1,y2,l2}));

    DLG2y2 = diff(LG2,Y2);
    dlg2y2=double(subs(DLG2y2,{Y1,Y2,L2},{y1,y2,l2}));

    DLG2l2 = diff(LG2,L2);
    dlg2l2=double(subs(DLG2l2,{Y1,Y2,L2,X1,X2},{y1,y2,l2,x1,x2}));
    G2 = [dlg2y1;dlg2y2;dlg2l2];

    % Hessiana System2
    H1y1 = diff(DLG2y1,Y1);
    h1y1=double(subs(H1y1,{Y1,Y2,L2},{y1,y2,l2}));
    H1y2 = diff(DLG2y1,Y2);
    h1y2=double(subs(H1y2,{Y1,Y2,L2},{y1,y2,l2}));
    H1l2 = diff(DLG2y1,L2);
    h1l2=double(subs(H1l2,{Y1,Y2,L2},{y1,y2,l2}));
    H21 = [h1y1,h1y2,h1l2];

    H2y1 = diff(DLG2y2,Y1);
    h2y1=double(subs(H2y1,{Y1,Y2,L2},{y1,y2,l2}));
    H2y2 = diff(DLG2y2,Y2);
    h2y2=double(subs(H2y2,{Y1,Y2,L2},{y1,y2,l2}));
    H2l2 = diff(DLG2y2,L2);
    h2l2=double(subs(H2l2,{Y1,Y2,L2},{y1,y2,l2}));
    H22 = [h2y1,h2y2,h2l2];

    H3y1 = diff(DLG2l2,Y1);
    h3y1=double(subs(H3y1,{Y1,Y2,L2},{y1,y2,l2}));
    H3y2 = diff(DLG2l2,Y2);
    h3y2=double(subs(H3y2,{Y1,Y2,L2},{y1,y2,l2}));
    H3l2 = diff(DLG2l2,L2);
    h3l2=double(subs(H3l2,{Y1,Y2,L2},{y1,y2,l2}));
    H23 = [h3y1,h3y2,h3l2];
    H2 = [H21;H22;H23];

    % Newton's Method System 2
    h2 = -inv(H2) * G2;
    Y = Y + h2'
    y1=Y(1,1);
    y2=Y(1,2);
    l2=Y(1,3);
```

```
    %% Convergence %%
    P  = [x1,x2,y1,y2];
    Pold = [x1old,x2old,y1old,y2old];
    E = P - Pold;
    e1 = sqrt ( E(1)^2 + E(2)^2 + E(3)^2 + E(4)^2);
    e2 = sqrt ( P(1)^2 + P(2)^2 + P(3)^2 + P(4)^2);
    erel = e1 / e2
    if erel < 10^(-4)
        'Problem has finished'
        break;
     end
end
```

APPENDIX B: Codes of Building of 8 zones

B.1 Dynamic for 8 zone Building Matlab code

Schoolmodel 8 zones

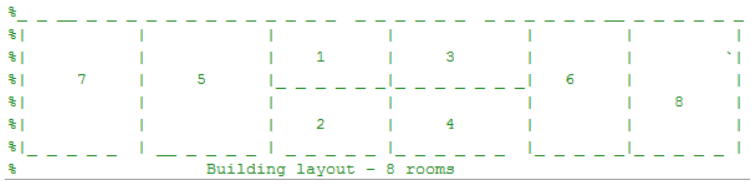
```
%%  variables declaration
nx=8;nu=8;nh=8;
x=sym('x', [nx 1]);     % states == temperature of each zone
u=sym('u', [nu 1]);     % input == supply air flow
lambda=sym('lambda', [nh,1]); % lagrange multiplier
q=sym('q', [nx 1]); % hea flux due to occupancy
syms Tsa Toa % supply air temperature
%% building properties
rhoaCpa=1.25625*1.005; % density of air * heat capacity of air
Uw1Aw1=5*9*0.0010/1000; % heat trnasfer rate for walls shared by both the
rooms (1/Rij from equation 2, systol paper)
Uw2Aw2=5*9*0.0010/1000; % heat trnasfer rate for walls shared by both the
rooms (1/Rij from equation 2, systol paper)
URAR=12*0.001/1000; % heat trnasfer rate for walls shared by both the
roof (1/Rext from equation 2, systol paper)
Cz=47.100; % heat capacity of zone/room (Ci in equation 2 )

h1=((u(1)*rhoaCpa*(Tsa-x(1)))+(2*Uw1Aw1*(x(2)-x(1)))+(2*Uw1Aw1*(x(5)-
x(1)))+(2*Uw2Aw2*(x(3)-x(1)))+q(1)+(URAR*(Toa-x(1))))/Cz; %energy balance
equation for room 1
h2=((u(2)*rhoaCpa*(Tsa-x(2)))+(2*Uw1Aw1*(x(1)-x(2)))+(2*Uw1Aw1*(x(5)-
x(2)))+(2*Uw2Aw2*(x(4)-x(2)))+q(2)+(URAR*(Toa-x(2))))/Cz;%energy balance
equation for room 2
h3=((u(3)*rhoaCpa*(Tsa-x(3)))+(2*Uw1Aw1*(x(1)-x(3)))+(2*Uw1Aw1*(x(6)-
x(3)))+(2*Uw2Aw2*(x(4)-x(3)))+q(3)+(URAR*(Toa-x(3))))/Cz; %energy balance
equation for room 3
h4=((u(4)*rhoaCpa*(Tsa-x(4)))+(2*Uw1Aw1*(x(3)-x(4)))+(2*Uw1Aw1*(x(6)-
x(4)))+(2*Uw2Aw2*(x(2)-x(4)))+q(4)+(URAR*(Toa-x(4))))/Cz;%energy balance
equation for room 4
h5=((u(5)*rhoaCpa*(Tsa-x(5)))+(2*Uw1Aw1*(x(2)-x(5)))+(2*Uw2Aw2*(x(1)-
x(5)))+(2*Uw1Aw1*(x(7)-x(5)))+q(5)+(URAR*(Toa-x(5))))/Cz; %energy balance
equation for room 5
h6=((u(6)*rhoaCpa*(Tsa-x(6)))+(2*Uw1Aw1*(x(3)-x(6)))+(2*Uw2Aw2*(x(4)-
x(6)))+(2*Uw1Aw1*(x(8)-x(6)))+q(6)+(URAR*(Toa-x(6))))/Cz; %energy balance
equation for room 6
h7=((u(7)*rhoaCpa*(Tsa-x(7)))+(2*Uw1Aw1*(x(5)-x(7)))+q(7)+(URAR*(Toa-
x(7))))/Cz; %energy balance equation for room 6
h8=((u(8)*rhoaCpa*(Tsa-x(8)))+(2*Uw1Aw1*(x(6)-x(8)))+q(8)+(URAR*(Toa-
x(8))))/Cz; %energy balance equation for room 6
h=[h1;h2;h3;h4;h5;h6;h7;h8];


%% Calculation of Jacobian matrix
A=jacobian(h,x);
B=jacobian(h,[u ;Tsa ;q  ;Toa ]);
C=jacobian(x,x);
D=jacobian(x,[u ;Tsa ;q  ;Toa ]);
```

```
%% finding operating point
Tsa=26; % supply air temperature Ts in equation 2
Toa=5; % outside/weather tempearture
x1=23;x2=23;x3=23;x4=23;x5=23;x6=23;x7=23;x8=23; % states OP initial
temperature of romm is 23 deg
u1=0.192;u2=0.192;u3=0.192;u4=0.192;u5=0.192;u6=0.192;u7=0.192;u8=0.192;
% input OP
q1=0.65;q2=0.65;q3=0.65;q4=0.65;q5=0.65;q6=0.65;q7=0.65;q8=0.65; % dist
OP
A1=eval(A);
B1=eval(B);
C1=eval(C);
D1=eval(D);
sys=c2d(ss(A1,B1,C1,D1),60);% sampling time 60sec for discretization
```

B.2 Centralized MPC Matlab code

Main program code

```
%%% MPC program for 8 zones with economic/energy minimization:
clc
yalmip('clear')
clear all
close all
```




```
%% import the dynamic mode for 8 zones- building
run('schoolmodel_3_8zones');
clearvars -except sys nx nu Tsa Toa
A=sys.a ;
Bu1=sys.b(:,1:nu);
Bu2=sys.b(:,nu+1);
Bd=sys.b(:,nu+2:end);
C=sys.c;
tsim=24*4*5;
run('generate_reference') % generate setpoint-comfort range
nd=nx+1;
%% MPC data
k1 = [1 1 1 1 1 1 1 1]*0.5;  % cost per kw for central supply fan
k2= 1; % cost per kj by central heating coil
Np =24;  % horizon as one day
uop=ones(nu,1)*0.192;

%% Controller code
%%  Initialization and variable declaration
uu = sdpvar(repmat(nu,1,Np),ones(1,Np));
xx = sdpvar(repmat(nx,1,Np),repmat(1,1,Np));
dd = sdpvar(repmat(nd,1,Np),repmat(1,1,Np));
zeta = sdpvar(1,1);
constraints = [];
objective = 0;
```

```
for k = 1:Np-1
 objective = objective
+k1*((uu{k}+uop).^2)+k2*sum(uu{k}+uop)+zeta;%+norm(R*(uu{k}+uop),2);

constraints = [constraints, xx{k+1} == A*xx{k} + Bu1*uu{k}+Bd*dd{k}];
constraints = [constraints, -0.192 <= uu{k}<= 0.31,-zeta<=xx{k}<=zeta, zeta <=
1,zeta >= -1];

end
parameters_in={xx{1},[dd{:}]};
solutions_out = {[uu{:}], [xx{:}],objective};
controller = optimizer(constraints,
objective,[],parameters_in,solutions_out);

%% MPC Control loop
x=zeros(nx,1);pastu=zeros(nu,1);
for t=1:tsim
% load disturbance
    [distp,distt,distds]=distpredicition8zone(t,Np);

% solving an ptimization problem
    [solutions,diagnostics] = controller{{x,distp}};
     if diagnostics == 1
        error('The problem is infeasible');
        break;
     end
     U = solutions{1}; X = solutions{2};

% Applying on building
  x=A*x+Bu1*U(:,1)+Bd*distt; %%  x=(dx+x);
  y=C*x+0.07*rand(nx,1);

% save data

UU(:,t)=U(:,1)+uop;
XX(:,t)=x+ones(nx,1)*23;
YY(:,t)=y+ones(nx,1)*23;
Dist(:,t)=distt+[ones(nx,1)*0.65 ;Toa];

end
xl=Ulb(1:tsim)+23;xu=Uub(1:tsim)+23;
```

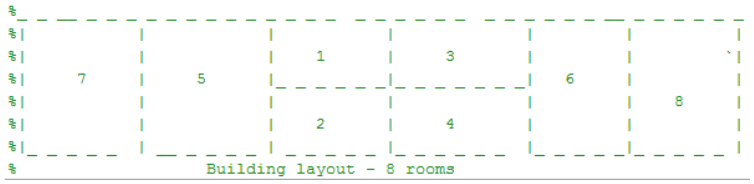
Reference trajectory

```
%% generate a refernce trajectory
clc
load('dist')% loading disturbace
l=length(dist(:,1));
for i=1:l
if (dist(i,1)>0)
    Ulb(i)=-0.5;Uub(i)=0.5;
else
    Ulb(i)=-8.5;Uub(i)=8.5;
end
end
```

B.3 Sensitivity-based coordination D-MPC Matlab code

Main program code

```
%%% MPC program for 8 zones with economic/energy minimization:
clc
yalmip('clear')
clear all
close all
```




```
%% import the dynamic mode for 8 zones- building
run('schoolmodel_3_8zones');
A=sys.a ;
Bu1=sys.b(:,1:nu);
Bu2=sys.b(:,nu+1);
Bd=sys.b(:,nu+2:end);
C=sys.c;
tsim=24*4*5;
run('generate_reference') % generate setpoint-comfort range
Np =24;  % horizon as one day
uop=ones(nx/2,1)*0.192;
k1 = ones(1,nx/2)*1;  % cost per kw for central supply fan
k2= 1; % cost per kj by central heating coil
k3= 20; %weight to supress oscillation

A11=[A(1:2,1:2) A(1:2,5) A(1:2,7);A(5,1:2) A(5,5)  A(5,7);A(7,1:2) A(7,5) A(7,7)];
A12=[A(1:2,3:4) A(1:2,6) A(1:2,7); A(5,3:4) A(5,6) A(5,8);A(7,3:4) A(7,6) A(7,8) ];
B1=[Bu1(1:2,1:2) Bu1(1:2,5)  Bu1(1:2,7);Bu1(5,1:2) Bu1(5,5) Bu1(5,7);Bu1(7,1:2) Bu1(7,5)
Bu1(7,7)];
Bd1=[Bd(1:2,1:2) Bd(1:2,5) Bd(1:2,7) Bd(1:2,9);Bd(5,1:2) Bd(5,5) Bd(5,7) Bd(5,9);Bd(7,1:2)
Bd(7,5) Bd(7,7) Bd(7,9)];
A22=[A(3:4,3:4) A(3:4,6)  A(3:4,8);A(6,3:4) A(6,6) A(6,8);A(8,3:4) A(8,6) A(8,8)];
A21=transpose(A12);
B2=[Bu1(3:4,3:4) Bu1(3:4,6)  Bu1(3:4,8);Bu1(6,3:4) Bu1(6,6) Bu1(6,8);Bu1(8,3:4) Bu1(8,6)
Bu1(8,8);];
Bd2=[Bd(3:4,3:4) Bd(3:4,6) Bd(3:4,8)  Bd(3:4,9);Bd(6,3:4) Bd(6,6) Bd(6,8) Bd(6,9);Bd(8,3:4)
Bd(8,6) Bd(8,8) Bd(8,9) ];

%% %%%%%%%%%%%%%%%%%%%%%%% controller for zone 1 %%%%%%%%%%%%%%%%%%%%%
%% decomposiiton into subsystems in overalapping structure
xl = sdpvar(1,1);
xu = sdpvar(1,1);
% Initial state
uu1 = sdpvar(repmat(nu/2,1,Np),ones(1,Np));
xx1 = sdpvar(repmat(nx/2,1,Np),repmat(1,1,Np));
dd1 = sdpvar(repmat(5,1,Np),repmat(1,1,Np));
zeta1=sdpvar(1,1);
u2old=sdpvar(4,1);
u1oldp=sdpvar(4,1);
constraints1 = [];
objective1 = 0;
xx12= sdpvar(4,1);
```

```
for k = 1:Np-1

objective1=objective1+k1*((uu1{k}+uop).^2)+k2*sum(uu1{k}+uop)+zeta1^2+k3*
norm((uu1{1}-u1oldp),1)+k3*norm((uu1{k+1}-uu1{k}),1);
constraints1 = [constraints1, xx1{k+1} == A11*xx1{k}+B1*uu1{k}+Bd1*dd1{k}+A12*xx12];
constraints1 = [constraints1, -0.192 <=uu1{k}<= 0.31,-zeta1+xl<=xx1{k}<=zeta1+xu, 0 <=
zeta1,zeta1 <= 0.5];
end
parameters_in1={xx1{1},[dd1{:}],u2old,xl,xu,xx12};
solutions_out1 = {[uu1{:}], [xx1{:}],objective1};
controller1 = optimizer(constraints1,
objective1,[],parameters_in1,solutions_out1);

%% %%%%%%%%%%%%%%%%% controller for zone 2%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Initial state
uu2 = sdpvar(repmat(nu/2,1,Np),ones(1,Np));
xx2 = sdpvar(repmat(nx/2,1,Np),ones(1,Np));
dd2 = sdpvar(repmat(5,1,Np),ones(1,Np));
zeta2=sdpvar(1,1);
u1old=sdpvar(4,1);
u2oldp=sdpvar(4,1);
constraints2 = [];
objective2 = 0;
xx34= sdpvar(4,1);

for k = 1:Np-1
objective2 = objective2 +
k1*((uu2{k}+uop).^2)+k2*sum(uu2{k}+uop)+zeta2^2+k3*norm((uu2{1}-
u2oldp),1)+k3*norm((uu2{k+1}-uu2{k}),1);
constraints2 = [constraints2, xx2{k+1} == A22*xx2{k}+B2*uu2{k}+Bd2*dd2{k}+A21*xx34];
constraints2 = [constraints2, -0.192 <= uu2{k}<= 0.31,-zeta2+xl<=xx2{k}<=zeta2+xu, 0 <=
zeta2,zeta2 <= 0.5 ];
end
parameters_in2={xx2{1},[dd2{:}],u1old,xl,xu,xx34};
solutions_out2 = {[uu2{:}], [xx2{:}],objective2};
controller2 = optimizer(constraints2,
objective2,[],parameters_in2,solutions_out2);

%%   %%%%%%%%%%%%%%%%%% Control starts here %%%%%%%%%%%%%%%%%%
xx=zeros(nx,1);
xx1=zeros(nx/2,1);
xx2=zeros(nx/2,1);
u1old=zeros(4,1);
u2old=zeros(4,1);
enerd=zeros(1,tsim);
for t=2:tsim
%% load disturbance
    [distp,distt,distds]=distpredicition8zone(t,Np);
%
%% solving an optimization problem for subsystem 1
    [solutions1,diagnostics1] = controller1{{xx1,distds,u2old,Ulb(t),Ulb(t),xx2}};
     if diagnostics1 == 1
        error('The problem is infeasible');
        break;
     end
     U1 = solutions1{1};
     X1 = solutions1{2};
     Z1=solutions1{3};
     u1old=(U1(:,1)+2*uop);
```

ETSEIB

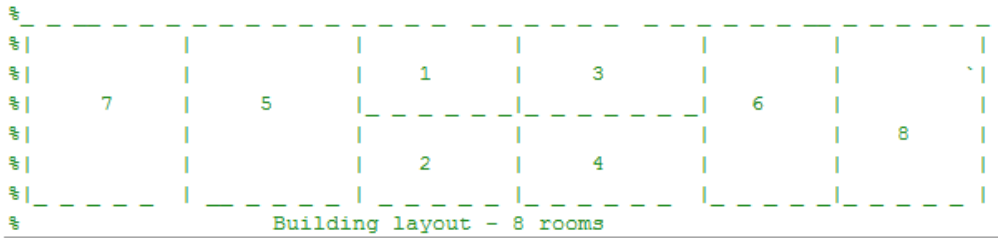
```
     u1oldp=U1(:,1);

%% solving an optimization problem for subsystem 2

    [solutions2,diagnostics2] = controller2{{xx2,distds,u1old,Ulb(t),Ulb(t),xx1}};
      if diagnostics2 == 1
         error('The problem is infeasible');
         break;
      end
      U2 = solutions2{1};
      X2 = solutions2{2};
      Z2=solutions2{3} ;
      u2old=(U1(:,1)+2*uop);

%% Applying on building

xx=A*xx+Bu1*[U1(1:2,1);U2(1:2,1);U1(3,1);U2(3,1);U1(4,1);U2(4,1)]+Bd*dist
t; %%  x=(dx+x);
  y=C*xx+0.007*rand(nx,1);

% % save data

UU1(:,t)=[U1(1:2,1);U2(1:2,1);U1(3,1);U2(3,1);U1(4,1);U2(4,1)]+[uop;uop];
XX(:,t)=xx+ones(nx,1)*23;
YY(:,t)=y+ones(nx,1)*23;
Dist(:,t)= distt+[ones(nx,1)*0.65 ;Toa];
enerd(:,t)=enerd(:,t-1)+sum((UU1(:,t)).^2)+sum(UU1(:,t));
xx1=[y(1:2);y(5);y(7)];
xx2=[y(3:4);y(6);y(8)];

 end
xl=Ulb(1:tsim)+22;xu=Uub(1:tsim)+24;
save('enerd','enerd');
```

Reference trajectory

```
%% generate a refernce trajectory
clc
load('dist')% loading disturbace
l=length(dist(:,1));
for i=1:l
if (dist(i,1)>0)
    Ulb(i)=-0.5;Uub(i)=0.5;
else
    Ulb(i)=-8.5;Uub(i)=8.5;
end
end
```

B.4 Optimal Conditions Decomposition D-MPC Matlab code

```
%%% MPC program for 8 zones with economic/energy minimization :
%%% Distributed OCD with state sinformation exchnage
clc
clear all
yalmip('clear')
close all
```




```
%% import the dynamic mode for 8 zones- building
run('schoolmodel_3_8zones');
A=sys.a ;
Bu1=sys.b(:,1:nu);
Bu2=sys.b(:,nu+1:2*nu+1);
Bd=sys.b(:,nu+2:end);
C=sys.c;
tsim=24*4*5;
run('generate_reference') % generate setpoint-comfort range
nd=nx+1;
Np =24;  % horizon as one day
uop=ones(nx/2,1)*23;
k1 = ones(1,nx/2)*1;  % cost per kw for central supply fan
k2= 1; % cost per kj by central heating coil
R = diag(ones(1,nu/2)*1); % decide imporatnce of inputs as chnage in
airflow will be at priority
%% decomposiiton into subsystems in overalapping structure
A11=[A(1:2,1:2) A(1:2,5) A(1:2,7);A(5,1:2) A(5,5)  A(5,7);A(7,1:2) A(7,5) A(7,8)];
A12=[A(1:2,3:4) A(1:2,6) A(1:2,7); A(5,3:4) A(5,6) A(5,8);A(7,3:4) A(7,6) A(7,8) ];
A21=[A(3:4,1:2) A(3:4,5) A(3:4,7); A(6,1:2) A(6,5) A(6,7);A(8,1:2) A(8,5) A(8,7)];
A22=[A(3:4,3:4) A(3:4,6)  A(3:4,8);A(6,3:4) A(6,6) A(6,8);A(8,3:4) A(8,6) A(8,8)];
B1T=[Bu1(1:2,1:2) Bu1(1:2,5)  Bu1(1:2,7);Bu1(5,1:2) Bu1(5,5) Bu1(5,7);Bu1(7,1:2) Bu1(7,5)
Bu1(7,7)];
B2T=[Bu1(3:4,3:4) Bu1(3:4,6)  Bu1(3:4,8);Bu1(6,3:4) Bu1(6,6) Bu1(6,8);Bu1(8,3:4) Bu1(8,6)
Bu1(8,8);];
B1F=[Bu2(1:2,1:2) Bu2(1:2,5)  Bu2(1:2,7);Bu2(5,1:2) Bu2(5,5) Bu2(5,7);Bu2(7,1:2) Bu2(7,5)
Bu2(7,7)];
B2F=[Bu2(3:4,3:4) Bu2(3:4,6)  Bu2(3:4,8);Bu2(6,3:4) Bu2(6,6) Bu2(6,8);Bu2(8,3:4) Bu2(8,6)
Bu2(8,8);];
Bd1=[Bd(1:2,1:2) Bd(1:2,5) Bd(1:2,7) Bd(1:2,9);Bd(5,1:2) Bd(5,5) Bd(5,7) Bd(5,9);Bd(7,1:2)
Bd(7,5) Bd(7,7) Bd(7,9)];
Bd2=[Bd(3:4,3:4) Bd(3:4,6) Bd(3:4,8)  Bd(3:4,9);Bd(6,3:4) Bd(6,6) Bd(6,8) Bd(6,9);Bd(8,3:4)
Bd(8,6) Bd(8,8) Bd(8,9) ];

xl = sdpvar(1,1);
xu = sdpvar(1,1);
```

```
%% %%%%%%%%%%%%%%%%% controller for zone 1 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Initial state
uu1 = sdpvar(repmat(nu/2,1,Np),ones(1,Np));
xx1 = sdpvar(repmat(nx/2,1,Np),repmat(1,1,Np));
lambda1 = sdpvar(repmat(3,1,Np-1),ones(1,Np-1));
dd1 = sdpvar(repmat(5,1,Np),repmat(1,1,Np));
zeta1=sdpvar(1,1);
u2old=sdpvar(4,1);
constraints1 = [];
objective1 = 0;
xx34= sdpvar(4,1);
uf1=ones(4,1)*0.192;
for k = 1:Np-1
 objective1=objective1+k1*((uu1{k}+uop).^2)+k2*sum(uu1{k}+uop)+zeta1^2;
 constraints1 = [constraints1, xx1{k+1} ==
A11*xx1{k}+B1T*uu1{k}+Bd1*dd1{k}+A12*xx34+B1F*uf1];
 constraints1 = [constraints1, -0.192 <=uu1{k}<= 0.31,-
zeta1+xl<=xx1{k}<=zeta1+xu, 0 <= zeta1,zeta1 <= 0.5];
end
parameters_in1={xx1{1},[dd1{:}],u2old,xx34,xl,xu};
solutions_out1 = {[uu1{:}], [xx1{:}],objective1};
controller1 = optimizer(constraints1,
objective1,[],parameters_in1,solutions_out1);

%% %%%%%%%%%%%%%%%%% controller for zone 2%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Initial state
uu2 = sdpvar(repmat(nu/2,1,Np),ones(1,Np));
xx2 = sdpvar(repmat(nx/2,1,Np),ones(1,Np));
dd2 = sdpvar(repmat(5,1,Np),ones(1,Np));
xx12= sdpvar(4,1);
zeta2=sdpvar(1,1);
u1old=sdpvar(4,1);
constraints2 = [];
objective2 = 0;
uf2=ones(4,1)*0.192;

for k = 1:Np-1
 objective2 = objective2 +
k1*((uu2{k}+uop).^2)+k2*sum(uu2{k}+uop)+zeta2^2;
constraints2 = [constraints2, xx2{k+1} ==
A22*xx2{k}+B2T*uu2{k}+Bd2*dd2{k}+A21*xx12+B2F*uf1];
constraints2 = [constraints2, -0.192 <= uu2{k}<= 0.31,-
zeta2+xl<=xx2{k}<=zeta2+xu, 0 <= zeta2,zeta2 <= 0.5 ];
end
parameters_in2={xx2{1},[dd2{:}],u1old,xx12,xl,xu};
solutions_out2 = {[uu2{:}], [xx2{:}],objective2};
controller2 = optimizer(constraints2,
objective2,[],parameters_in2,solutions_out2);

%%%%%%%%%%%%%%%%% Control start here %%%%%%%%%%%%%%%%%%%%%%%%%%%%%
xx=zeros(nx,1);
xx1=zeros(nx/2,1);
xx2=zeros(nx/2,1);
xx12=zeros(nx/2,1);
xx34=zeros(nx/2,1);
u1old=zeros(4,1);
```

```
u2old=zeros(4,1);
enerd=zeros(1,tsim);
for t=2:tsim
%% load disturbance
    [distp,distt,distds]=distpredicition8zone(t,Np);
%
%% solving an optimization problem for subsystem 1
    [solutions1,diagnostics1] =
controller1{{xx1,distds,u2old,xx34,Ulb(t),Uub(t)}};
     if diagnostics1 == 1
        error('The problem is infeasible');
        break;
     end
     U1 = solutions1{1};
     X1 = solutions1{2};
    [solutions2,diagnostics2] = controller2{{xx2,distds,u1old,xx12,Ulb(t),Uub(t)}};
     if diagnostics2 == 1
        error('The problem is infeasible');
        break;
     end
     U2 = solutions2{1};
     X2 = solutions2{2};

%% Applying on building

xx=A*xx+sys.b(:,1:2*nu)*[U1(1:2,1);U2(1:2,1);U1(3,1);U2(3,1);U1(4,1);U2(4
,1);uf1;uf2]+Bd*distt; %%  x=(dx+x);
  y=C*xx+0.007*rand(nx,1);

%% save data

UU1(:,t)=[U1(1:2,1);U2(1:2,1);U1(3,1);U2(3,1);U1(4,1);U2(4,1)]+[uop;uop];
XX(:,t)=xx+ones(nx,1)*23;
YY(:,t)=y+ones(nx,1)*23;
Dist(:,t)= distt+[ones(nx,1)*0.65 ;Toa];
enerd(:,t)=enerd(:,t-1)+sum((UU1(:,t)).^2)+sum(UU1(:,t));
xx1=[y(1:2);y(5);y(7)];
xx2=[y(3:4);y(6);y(8)];

 end
xl=Ulb(1:tsim)+22;
xu=Uub(1:tsim)+24;
save('enerd','enerd');
```