



Programme de simulation pour domotique

Promoteurs : M. C. Crincket
M. W. Claes

Aerts Thomas
Leroy Stijn
2006-2007

Préface

Ce travail de fin d'études est le résultat de beaucoup de recherches et de beaucoup de travail. Sans l'aide de certaines personnes, il est sûr que la réalisation de ce document n'aurait pas pu se faire facilement. C'est pourquoi nous voudrions remercier tous ceux qui ont contribué à la réalisation de celui-ci.

Tout d'abord, nous aimerions remercier M. C. Crincket, notre directeur du stage, pour ses conseils et ses suggestions.

Aussi nous voudrions remercier M. W. Claes, notre promoteur, pour ses conseils et sa motivation.

Puis nous voudrions remercier German Gutierrez Zubizarreta, pour son aide avec l'AutoCAD et son aide avec les dessins.

Nous voudrions également remercier les autres étudiants pour leur support moral, mais surtout nos parents qui nous avons donné l'opportunité d'étudier à la KHLim et faire un stage Erasmus.

Thomas Aerts & Stijn Leroy

Préface	2
1 Introduction	6
2 AutoCAD	7
2.1 Introduction	7
2.2 Calques (Layers)	7
2.3 Bloc (Block)	10
2.4 Aspects	12
3 Visual Basic.....	17
3.1 Eléments utilisés	17
3.1.1 Label	17
3.1.2 Command button	17
3.1.3 Scroll-bar	17
3.1.4 Cadre avec options	17
3.1.5 TextBox	18
3.1.6 PictureBox	18
3.1.7 Image	18
3.2 Le programme.....	19
3.2.1 Choix des langues.....	20
3.2.2 Propre programme.....	22
3.2.3 Ajuster la consigne	25
3.2.4 Bouton Exit	27
3.2.5 Choix de la force du vent.....	28
3.2.6 Choix de la clarté.....	29
3.2.7 Choix du régulateur	30
3.2.8 Le Timer	31
3.2.9 Le régulateur PID	33
3.2.10 Le régulateur On/Off.....	34
3.2.11 Module avec les variables.....	38
4 Régulateurs.....	39
4.1 Régulateur On/Off avec hystérésis	39
4.2 Régulateur PID	41
4.2.1 Action proportionnelle.....	41
4.2.2 Action intégrante	49
4.2.3 Action différentielle	49
4.2.4 Combinaison : PID	49
5 Bibliographie.....	51
6 Annexe 1: Le programme.....	52

Figure 1: créer un layer.....	7
Figure 2: propriétés d'un layer.....	7
Figure 3: couleur d'un layer	8
Figure 4: type de ligne.....	8
Figure 5: épaisseur de ligne	9
Figure 6: liste des layers utilisés	9
Figure 7: insérer un block	10
Figure 8: trouver le block.....	10
Figure 9: le block choisi	11
Figure 10: la maison en 3D.....	12
Figure 11: la façade de la maison.....	12
Figure 12: l'arrière de la maison.....	13
Figure 13: la côté droite de la maison	13
Figure 14: la côté gauche de la maison.....	13
Figure 15: vue aérienne	14
Figure 16: la maison avec les lumières.....	15
Figure 17: la maison avec les stores en bas	15
Figure 18: la maison avec les lumières et les stores.....	16
Figure 19: un label.....	17
Figure 20: un command button	17
Figure 21: un scroll-bar	17
Figure 22: cadre avec options	17
Figure 23: TextBox	18
Figure 24: PictureBox.....	18
Figure 25: Image	18
Figure 26: choix du fonctionnalité	19
Figure 27: Formulaire: frmTaal	20
Figure 28: Formulaire: frmProject	22
Figure 29: le bouton +5	25
Figure 30: Le bouton -5	25
Figure 31: scroll-bar pour le température	26
Figure 32: Le bouton "exit"	27
Figure 33: scroll-bar pour la force du vent	28
Figure 34: choix pour la clarté	29
Figure 35: choix pour la nuit.....	29
Figure 36: choix pour PID	30
Figure 37: variables pour le PID	30
Figure 38: choix pour le On/Off.....	31
Figure 39: hystérèse.....	31
Figure 40 : Graphique du régulateur On/Off	39
Figure 41: Principe du régulateur PID.....	41
Figure 42: boucle de régulation.....	41
Figure 43: réponse échelon du régulateur P.....	41
Figure 44: root-locus d'un procédé stable.....	42
Figure 45: réponse échelon d'un procédé stable.....	43
Figure 46: root-locus d'un procédé instable	44
Figure 47: réponse échelon d'un procédé instable.....	44
Figure 48: root-locus d'un procédé avec fortification critique.....	48
Figure 49: réponse échelon d'un procédé avec fortification critique	48

Figure 50: réponse échelon du régulateur I	49
Figure 51: réponse échelon du régulateur PD	49
Figure 52: réponse échelon théorique du PID.....	50
Figure 53: réponse échelon pratique du PID.....	50

1 Introduction

Notre mission a consisté en l'automatisation d'une maison. Nous avons conçu la maison avec le logiciel de AutoCAD et un programme écrit dans Visual Basic.

Dans la première partie, la construction de la maison est expliquée et une courte introduction est donnée au sujet de AutoCAD.

Ensuite nous présentons le programme réalisé. Nous donnons également une courte introduction au sujet de Visual Basic, ici les termes les plus importants sont présentés.

Les correcteurs utilisés sont expliqués pour finir en détail.

Notre intention était de réaliser un programme de simulation pour montrer la différence parmi un régulateur conventionnel, comme un thermostat et un régulateur PID, le type le plus souvent utilisé dans l'industrie.

Nous avons choisi pour ce projet à cause de notre intérêt pour l'automation et les régulateurs, mais aussi parce que nous aimons réaliser des programmes. Autre était pour montrer que les régulateurs conventionnels ne sont pas toujours les mieux.

2 AutoCAD

2.1 Introduction

AutoCAD est une suite de CAD software pour dessiner en 2- et 3-dimension. C'est un programme qui utilise des vecteurs pour construire des dessins. Il utilise des primitives de base, comme des lignes, des cercles, des arcs, et texte pour faire des objets complexes.

2.2 Calques (Layers)

Les calques sont des très bonnes aides pour organiser les propriétés. Avec les calques, c'est possible pour distinguer les fonctions à un dessin.

Il est possible de changer des propriétés des calques comme la couleur, le type ligne, épaisseur de ligne et le style d'impression...

Créer un nouveau calque en AutoCAD.

Nous sélectionnons l'outil *layer* dans le déroulant *format*.



Figure 1: créer un layer

Et nous choisissons le bouton *New*.

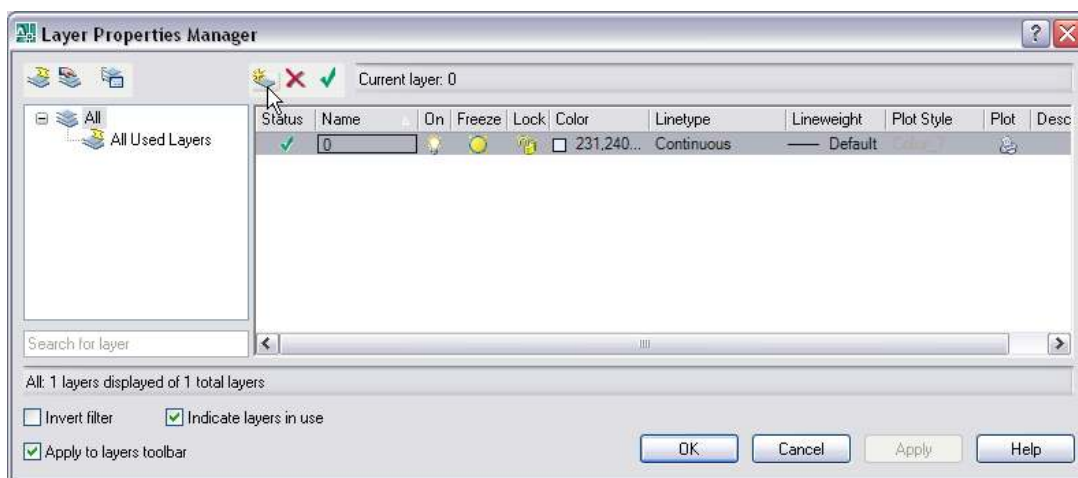


Figure 2: propriétés d'un layer

Nous choisissons un nom pour le nouveau layer et les propriétés, comme la couleur.

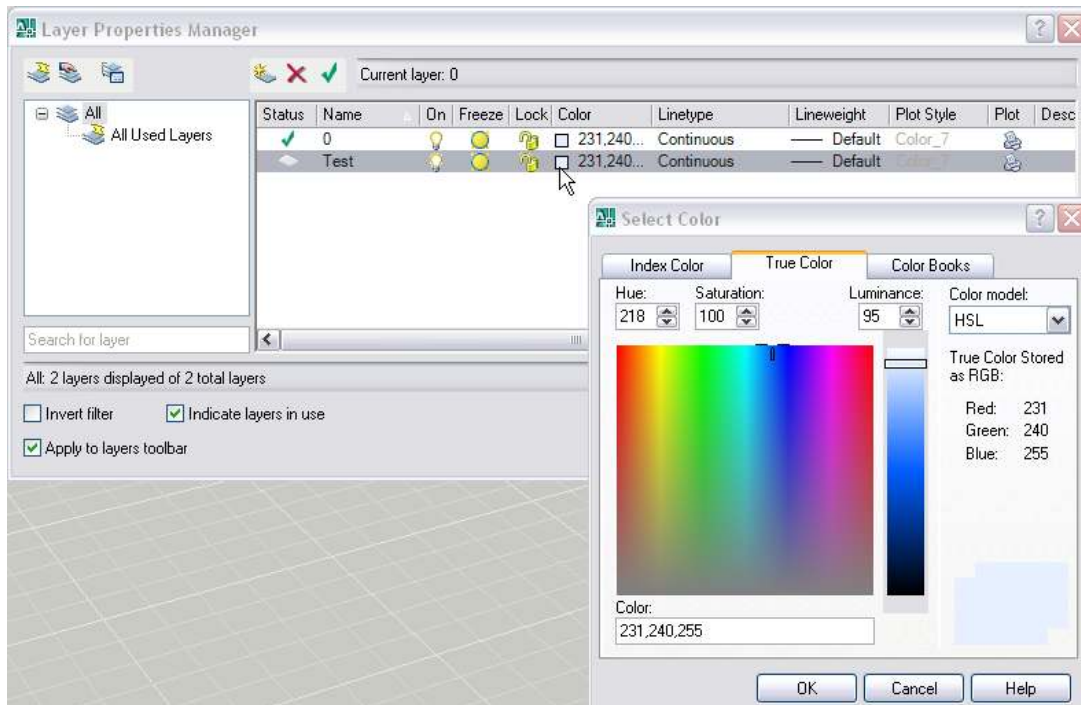


Figure 3: couleur d'un layer

C'est aussi possible de changer le *Linetype*. Nous cliquons sur *Continuous*. Dans le nouveau cadre nous choisissons le bouton *load* et maintenant c'est possible de choisir les lignes standard.

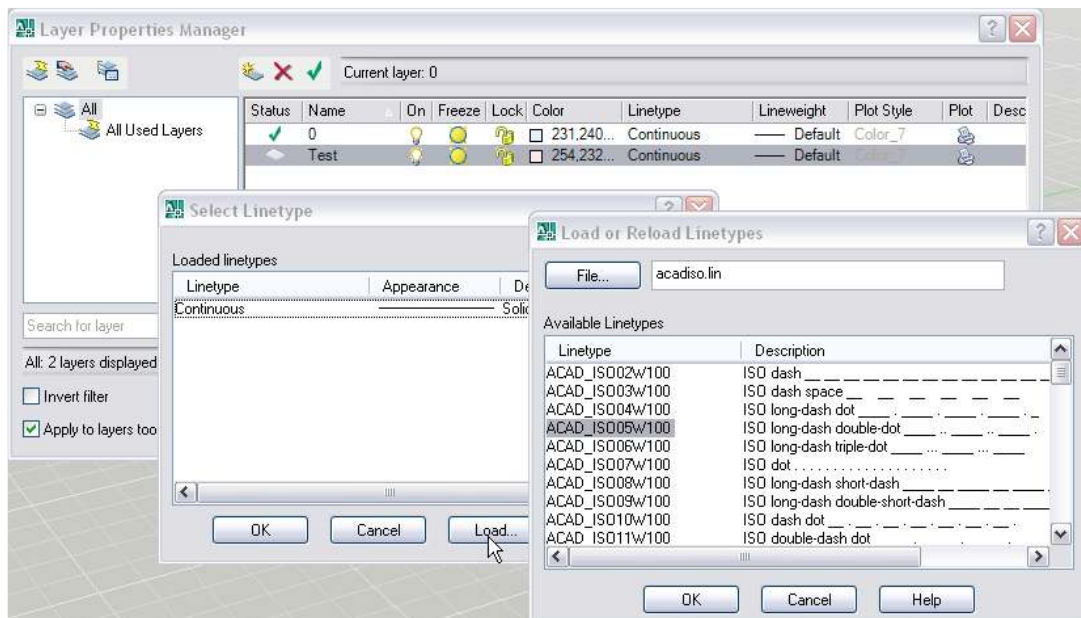


Figure 4: type de ligne

Pour changer l'épaisseur de ligne, nous cliquons sur *lineweight*.

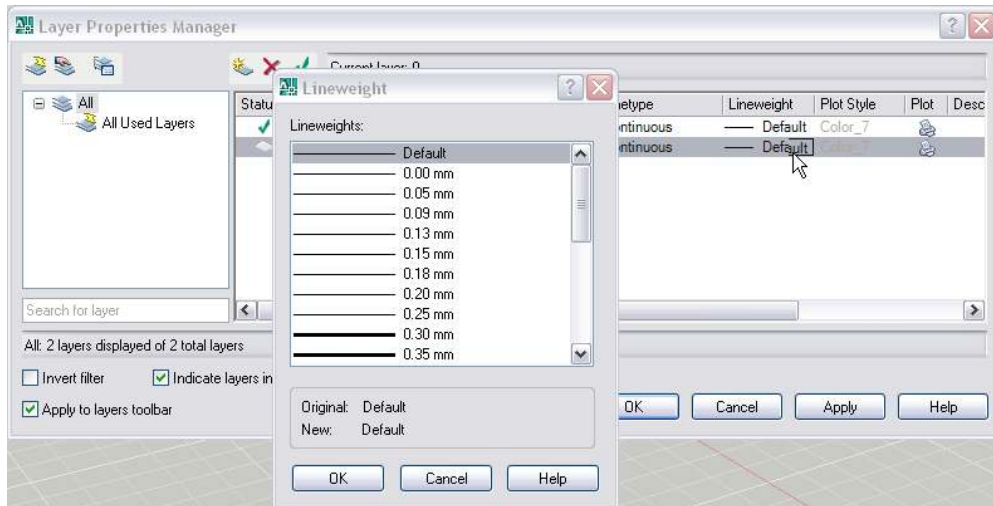


Figure 5: épaisseur de ligne

Pour tout les éléments il y a une autre layer, vous trouvez nos calques du projet en bas.

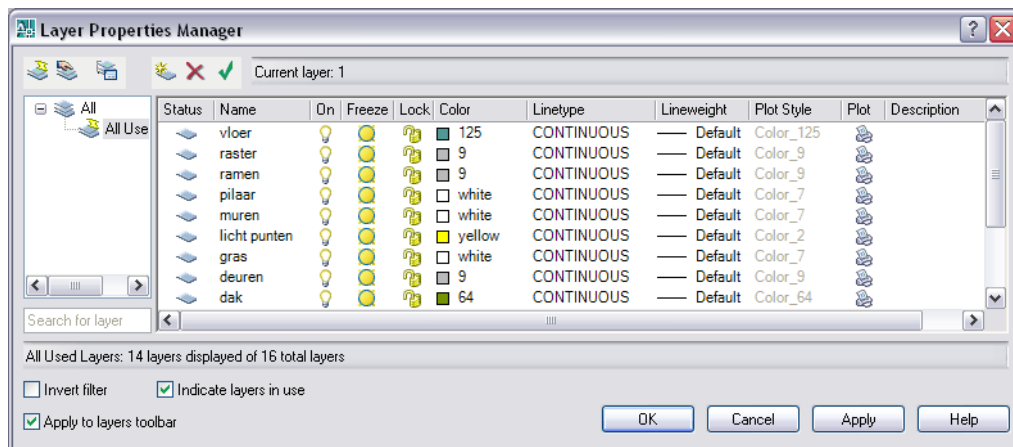


Figure 6: liste des layers utilisés

Traduction des calques :

vloer	sol
raster	grillage
ramen	fenêtres
pilaar	pilier
muren	murs
licht punten	source de lumières
gras	herbe
deuren	portes
dak	toit

On a choisi ces calques pour faciliter le dessin, on a des calques différents pour les murs, pour le toit, etc., parce que ils ont tous des différentes propriétés. C'est pourquoi on a décidé de diviser le dessin en des parties de base, un calque pour les fenêtres, autre pour le sol etc.

2.3 Bloc (Block)

Avec les *blocks*, vous pouvez gagner du temps parce que vous groupez les parties dans le dessin et lui-même ne dessine pas de nouveau.

La plantation est un block dans notre project.

Pour insérer un block, vous choisissez en les menu *Insert*, et choisissez *Block*.



Figure 7: insérer un block

Au cadre suivant, nous cliquons *Browse* pour choisir un *Block* existé.

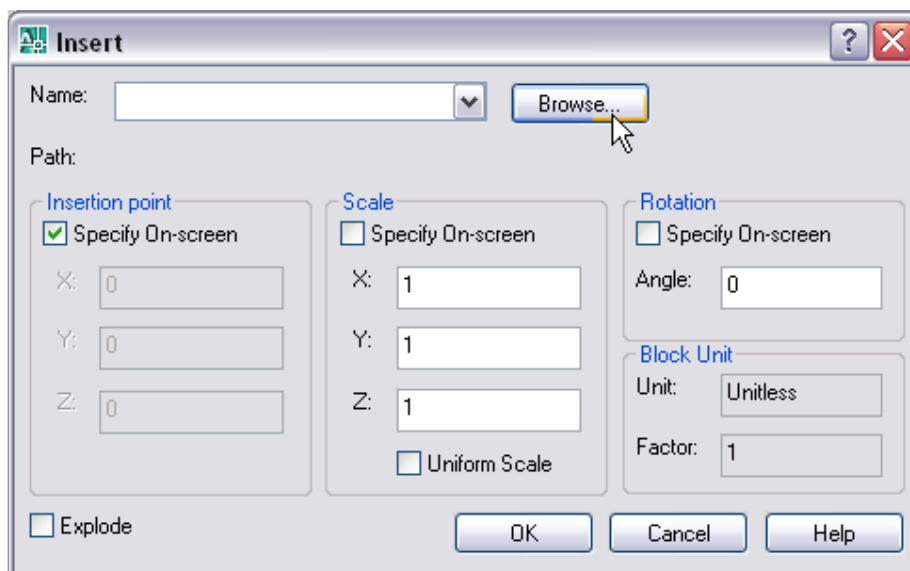


Figure 8: trouver le block

Par exemple le block suivant.

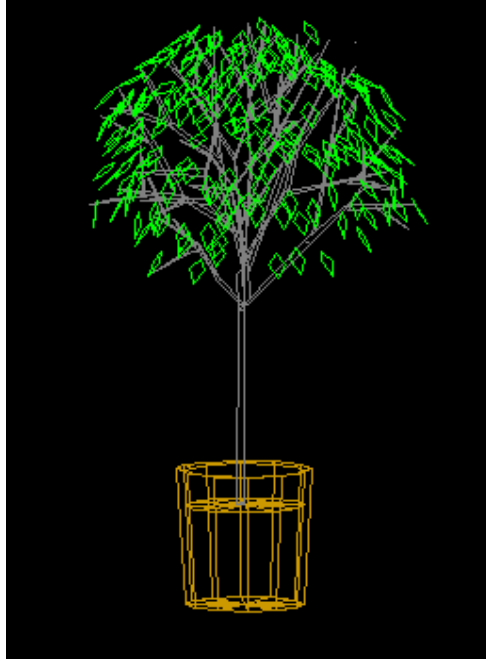


Figure 9: le block choisi

Et nous plaçons le block à la place désirable.

2.4 Aspects

En trois dimensions.

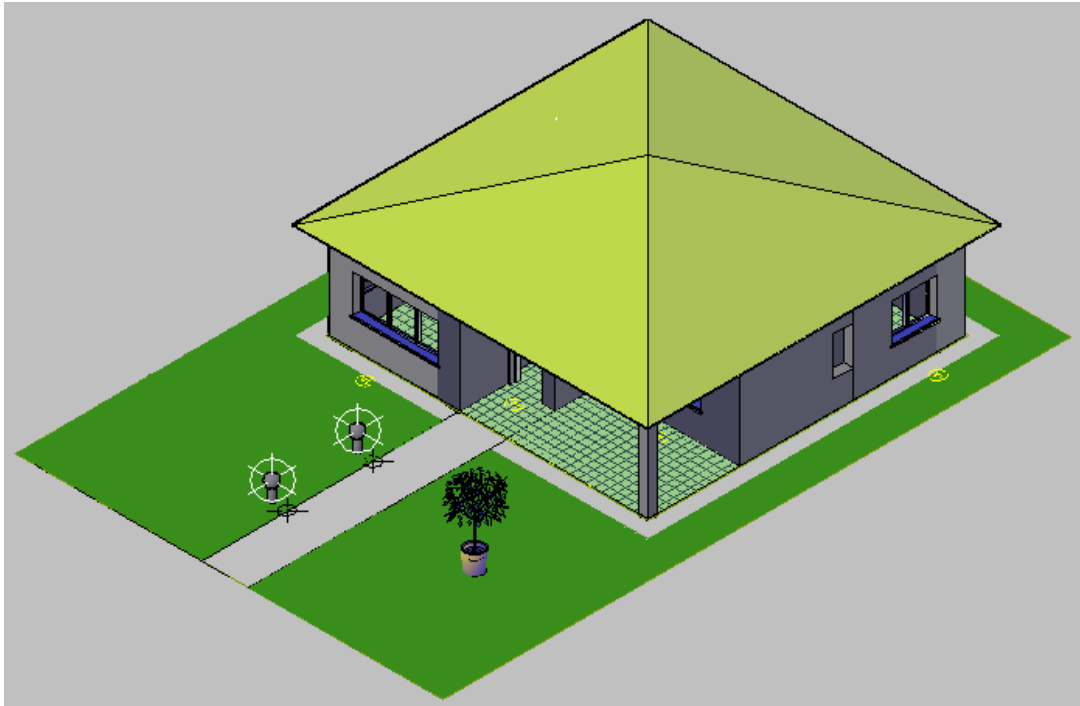


Figure 10: la maison en 3D

Façade

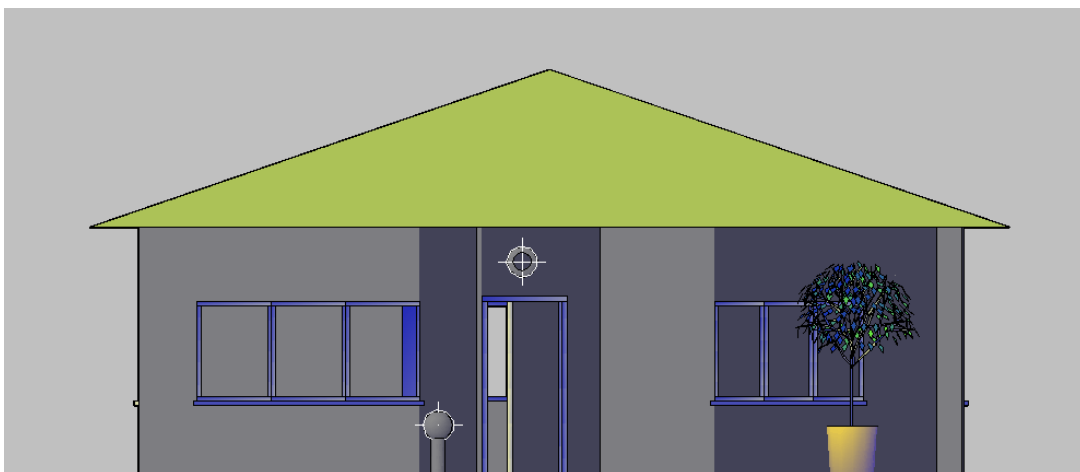


Figure 11: la façade de la maison

Arrière

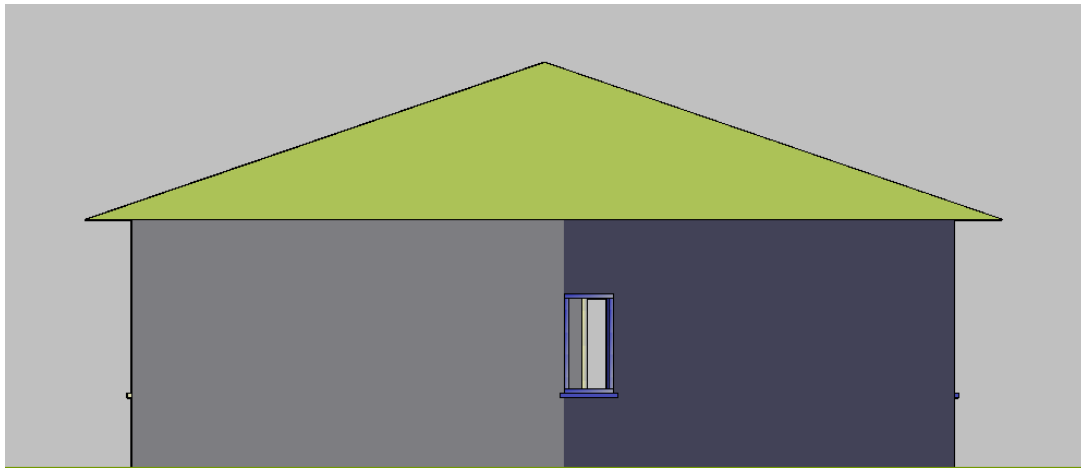


Figure 12: l'arrière de la maison

Côté droit

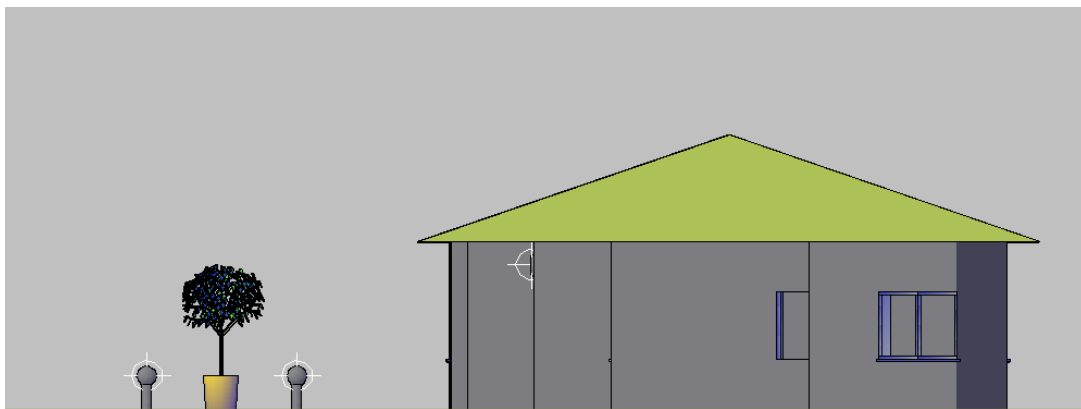


Figure 13: la côté droite de la maison

Côté gauche

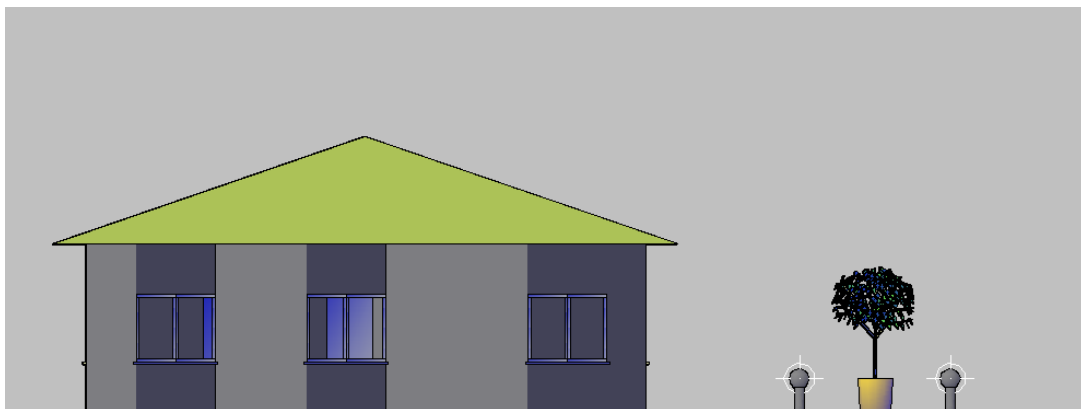


Figure 14: la côté gauche de la maison

Vue aérienne, de dessus

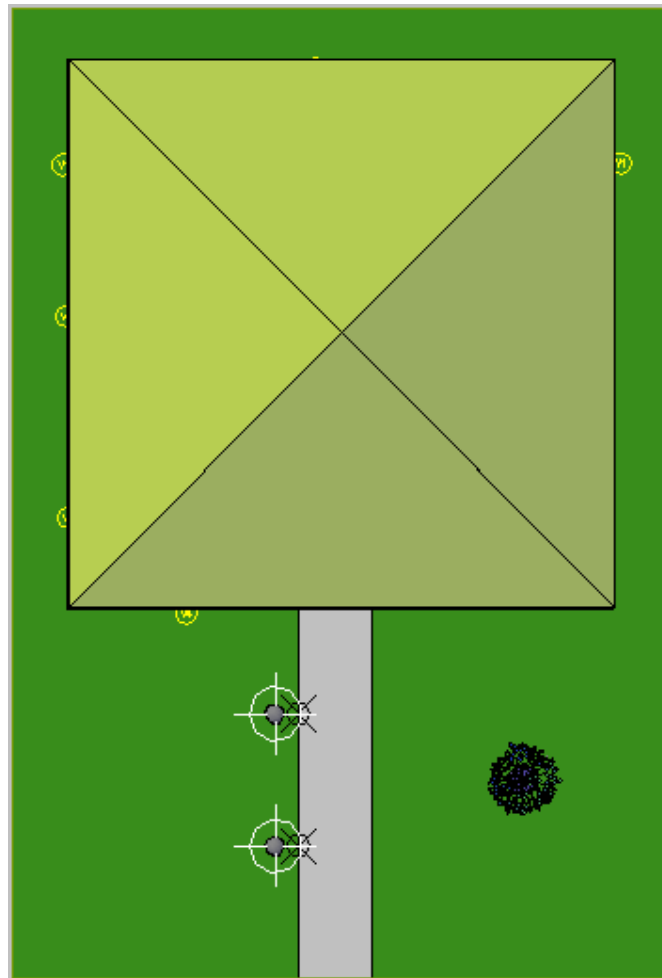


Figure 15: vue aérienne

Avec les lumières

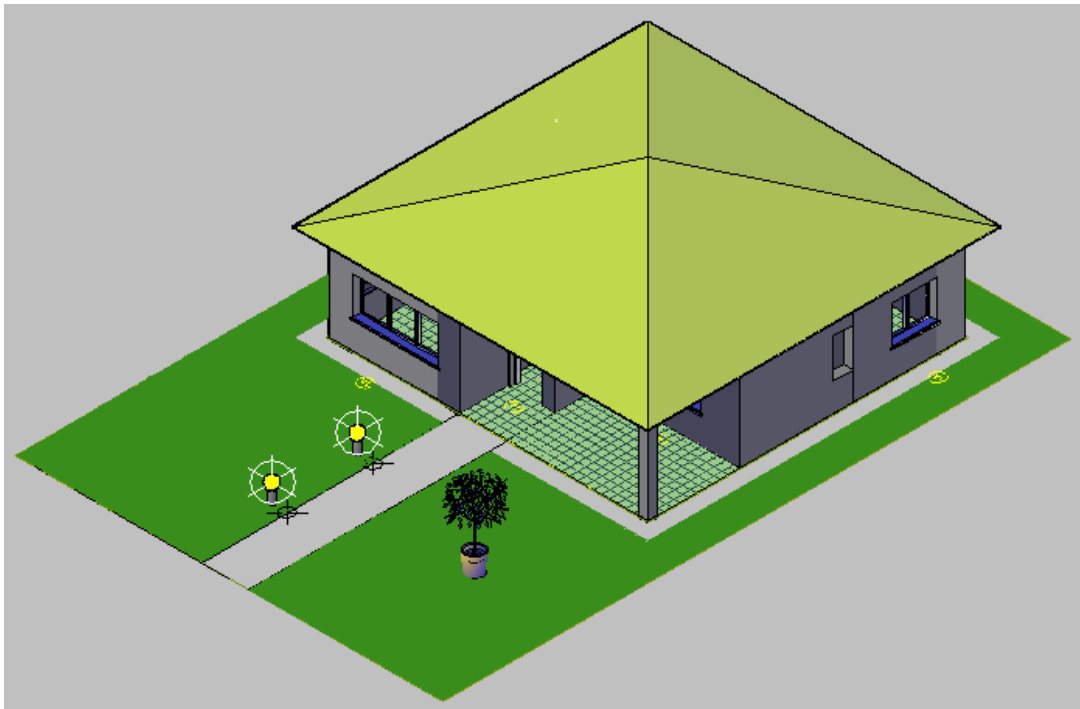


Figure 16: la maison avec les lumières

Avec les stores.

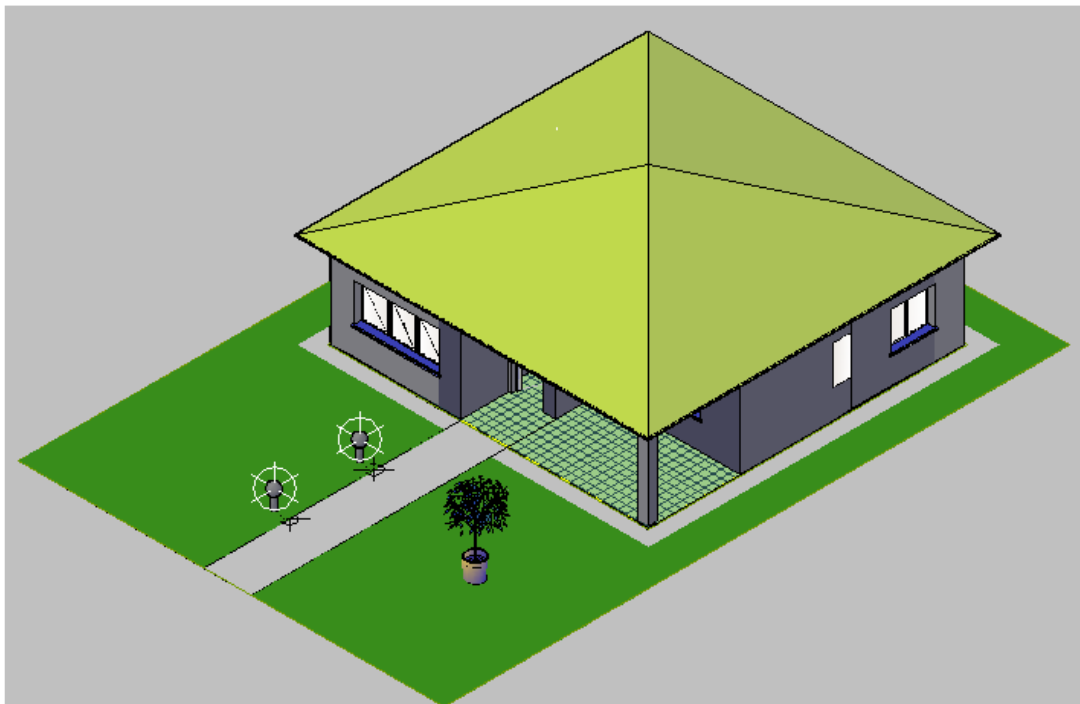


Figure 17: la maison avec les stores en bas

Avec les stores et les lumières.

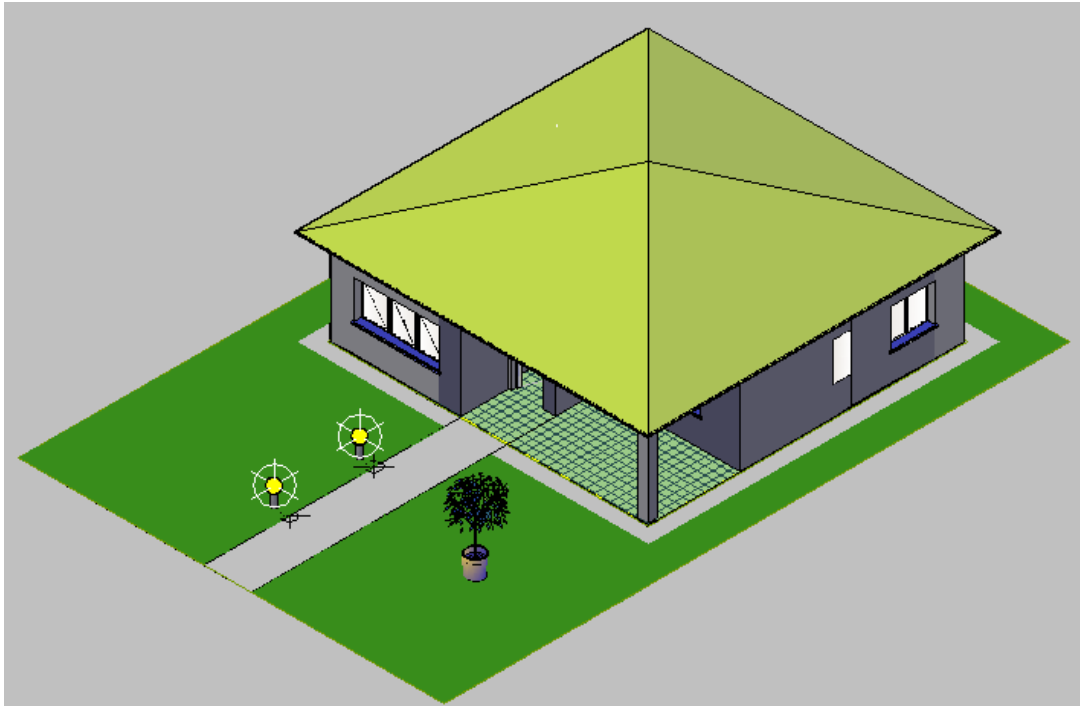


Figure 18: la maison avec les lumières et les stores

On a choisi pour dessiner la maison à cette manière parce on a besoin des fenêtres visibles pour visualiser les stores. Aussi on a besoin d'un jardin simple pour simuler les lumières quand il fait nuit.

3 Visual Basic

Le Visual Basic est une évolution des Basics précédents de Microsoft, qui permet de créer des applications fenêtrées et de pratiquer également la programmation événementielle. Bénéficiant de la simplicité du Basic originel, il permet de créer des programmes relativement rapidement. En Visual Basic on manipule des éléments visuels à l'écran auxquels il ne reste plus qu'à associer du code avec des boîtes de dialogue.

3.1 Éléments utilisés

3.1.1 Label

Un « label » est utilisé de montrer du texte qui ne peut pas être changer par l'utilisateur.

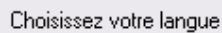


Figure 19: un label

3.1.2 Command button

Le « command button » exécute une certaine action quand l'utilisateur l'appuie.



Figure 20: un command button

3.1.3 Scroll-bar

Avec un « scroll-bar » on change la valeur annexe.



Figure 21: un scroll-bar

3.1.4 Cadre avec options

Un cadre (Frame) est utilisé de grouper des autres commandes. Comme les options.




Figure 22: cadre avec options

3.1.5 TextBox

Un « TextBox » est utilisé pour donner ou pour montrer l'information.



Figure 23: TextBox

3.1.6 PictureBox

Dans un « PictureBox » on peut montrer des bitmaps, des icônes ou des Windows metafiles. On peut aussi l'utiliser pour placer du texte ou grouper des commandes. Nous l'avons programmé comme animé.

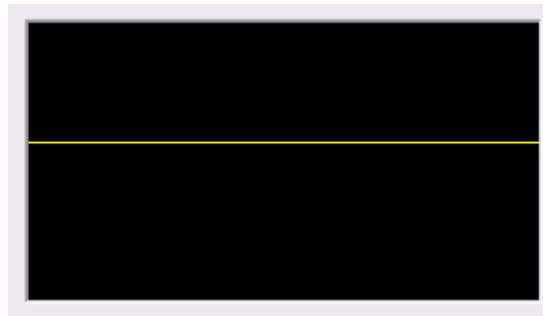


Figure 24: PictureBox

3.1.7 Image

Avec « image » on peut montrer des bitmaps, des icônes ou des Windows metafiles. On peut aussi utiliser un « image » comme « command button ».



Figure 25: Image

3.2 Le programme

Parce que on a deux fonctionnalités, on doit choisir lequel met en marche le premier

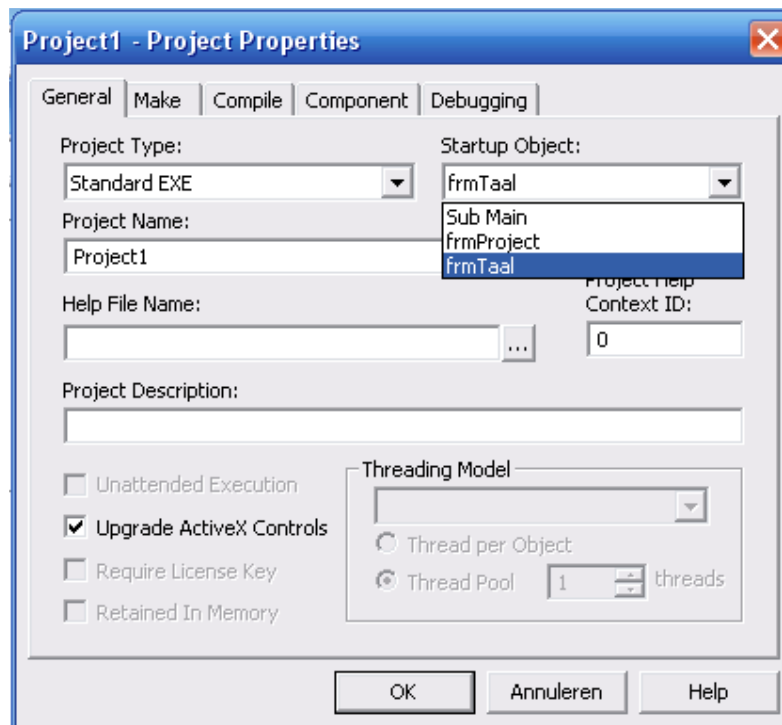


Figure 26: choix du fonctionnalité

3.2.1 Choix des langues

Quand on met en marche le programme on voit l'écran suivant. Ici on peut choisir la langue pour communiquer avec le programme, l'anglais, le français, le néerlandais ou l'allemand.

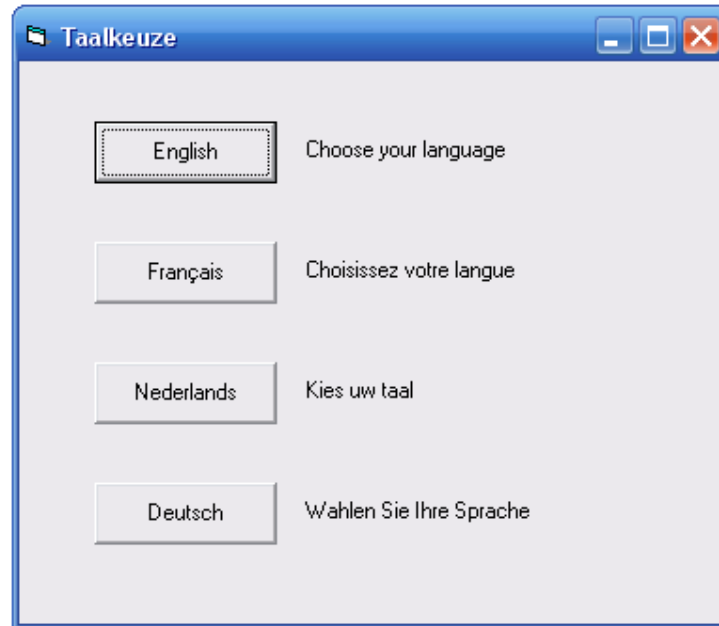


Figure 27: Formulaire: frmTaal

Le code de ce formulaire est le suivant :

```
Private Sub formLoad()
```

```
'Avec le formload() on peut établir ce que doit arriver,
```

```
'quand le formulaire devient charger.
```

```
'Avec .Hide on cache un objet, dans ce cas le formulaire "frmProject"
```

```
'Avec .Show on montre un objet, dans ce cas le formulaire "frmTaal"
```

```
frmProject.Hide
```

```
frmTaal.Show
```

```
End Sub
```

```
Private Sub cmdDuits_Click()
```

```
'Avec _Click() on programme une action qui arrive quand, on appuie sur cet objet.
```

```
'Le valeur de la variable "intTaal" est une indication pour la langue choisi.
```

```
'intTaal=1 -> anglais
```

```
'intTaal=2 -> français
```

```
'intTaal=3 -> néerlandais
```

```
'intTaal=4 -> allemand
```

```
intTaal = 4
```

```
frmTaal.Hide
```

```
frmProject.Show  
End Sub
```

```
Private Sub cmdEngels_Click()  
intTaal = 1  
frmTaal.Hide  
frmProject.Show  
End Sub
```

```
Private Sub cmdFrans_Click()  
intTaal = 2  
frmTaal.Hide  
frmProject.Show  
End Sub
```

```
Private Sub cmdNederlands_Click()  
intTaal = 3  
frmTaal.Hide  
frmProject.Show  
End Sub
```

3.2.2 Propre programme

Et le fonctionnement du propre programme est le suivant:



Figure 28: Formulaire: frmProject

Quand on met en marche un programme la première chose qui est exécutée est le « Form_load() »

```
Private Sub Form_load()
```

```
' Avec le formload() on peut établir ce que doit arriver,
```

```
' quand le formulaire devient charger.
```

```
' Pour centrer le formulaire:
```

```
frmProject.Left = (Screen.Width / 2) - (frmProject.Width / 2)
```

```
frmProject.Top = (Screen.Height / 2) - (frmProject.Height / 2)
```

```
' Pour charger les images standard (LoadPicture):
```

```
imgDoorsnede.Picture = LoadPicture(App.Path & "\\verwarming_aan_licht_uit.bmp")
```

```
img3d.Picture = LoadPicture(App.Path & "\\standaard.bmp")
```

```
' Avec la variable "intTaal" d'autre formulaire "frmTaal" on a choisi la langue  
' du programme
```

```
' Avec .Caption on peut ajuster le texte sur les "labels", parce que c'est devant  
' le signe d'égalité' la structure "Select Case - End Select":
```

```
' Select Case <variable>
```

```
' Case <x1>
```

```
' <action(s)>
```

```
' Case <x2>
```

```

'      <action(s)>
'
'      ...
' End Select
Select Case intTaal
Case 1
    lblWind.Caption = "Wind"
    lblSetpoint.Caption = "Desired temperature:"
    fraLicht.Caption = "Light:"
    optDuister.Caption = "Dark"
    optLicht.Caption = "Light"
    fraRegelaar.Caption = "Regulator:"
    lblTemp.Caption = "Current temperature:"

Case 2
    lblWind.Caption = "Vent"
    lblSetpoint.Caption = "Température désirée:"
    fraLicht.Caption = "Clarté:"
    optDuister.Caption = "Nuit"
    optLicht.Caption = "Jour"
    fraRegelaar.Caption = "Régulateur:"
    lblTemp.Caption = "Température présente:"

Case 3
    lblWind.Caption = "Wind"
    lblSetpoint.Caption = "Gewenste temperatuur:"
    fraLicht.Caption = "Lichtsterkte:"
    optDuister.Caption = "Donker"
    optLicht.Caption = "Licht"
    fraRegelaar.Caption = "Regelaar:"
    lblTemp.Caption = "Huidige temperatuur:"

Case 4
    lblWind.Caption = "Wind"
    lblSetpoint.Caption = "Gewünschte Temperatur:"
    fraLicht.Caption = "Licht:"
    optDuister.Caption = "Dunkel"
    optLicht.Caption = "Licht"
    fraRegelaar.Caption = "Regler"
    lblTemp.Caption = "Heutige Temperatur:"

End Select

' Maintenant on établit la valeur des variables standard:
' Ici avec .Value on ajuste la valeur d'objet,
' parce c'est devant le signe d'égalité
hscSetpoint.Value = 23

' Ici on utilise .value pour mettre en mémoire le consigne,
' parce c'est après le signe d'égalité
intSetpoint = hscSetpoint.Value

```

```
lblPv.Caption = CInt(dblProceswaarde) & "°C"
```

```
intWaarde = 2000
```

```
intToevoer = (10 * (intWaarde / 100)) / 6
```

```
optPid.Value = True
```

```
optLicht.Value = True
```

```
' Pour mettre en mémoire les valeurs des variables du régulateur:
```

```
' Ici on utilise .Text pour donner la valeur de "TextBox" au variable,
```

```
' parce que c'est après le signe d'égalité
```

```
dblKr = txtKr.Text
```

```
dblTi = txtTi.Text
```

```
dblTd = txtTd.Text
```

```
' L'option pour le régulateur est standard l'option de régulateur PID,
```

```
' et avec .Visible on peut cacher ou montrer un objet dépendant de la
```

```
' valeur de l'expression: "True" (vrai) ou "False" (faux)
```

```
If optPid.Value = True Then
```

```
    lblTd.Visible = True
```

```
    txtTd.Visible = True
```

```
    lblTi.Visible = True
```

```
    txtTi.Visible = True
```

```
    lblKr.Visible = True
```

```
    txtKr.Visible = True
```

```
    lblHysteresis.Visible = False
```

```
    txtHysteresis.Visible = False
```

```
End If
```

```
' Pour établir les propriétés de "PictureBox" pour montrer le graphique
```

```
pctGrafiek.Cls
```

```
pctGrafiek.ScaleMode = 3
```

```
pctGrafiek.ScaleHeight = 110
```

```
pctGrafiek.ScaleWidth = 100
```

```
pctGrafiek.AutoRedraw = True
```

```
pctGrafiek.ForeColor = vbCyan
```

```
pctGrafiek.DrawStyle = 0
```

```
pctGrafiek.DrawWidth = 1
```

```
End Sub
```


3.2.3 Ajuster la consigne

On a trois manières pour ajuster la consigne :

1. Avec la bouton « +5 »
2. Avec la bouton « -5 »
3. Avec le scroll-bar

Les boutons de 5 en 5 serrent pour donner un changement immédiat et clairement visible. Avec ces boutons on est en état de simuler une réponse échelon. Ensuite on peut voir comment la procède réagit.

Le bouton « +5 » :

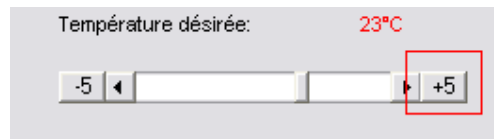


Figure 29: le bouton +5

Le code :

```
Private Sub cmdPlus5_Click()  
' Quand on appuie ce bouton ajoute la consigne est ajouté avec 5  
intSetpoint = intSetpoint + 5  
' Le boucle "If - End If":  
' If <condition(s)> Then  
'   <action(s)>  
' End If  
' Pour mettre la limite supérieur (40°C) à la consigne:  
If intSetpoint > 40 Then  
    intSetpoint = 40  
End If  
hscSetpoint.Value = intSetpoint  
End Sub
```

Le bouton « -5 » :

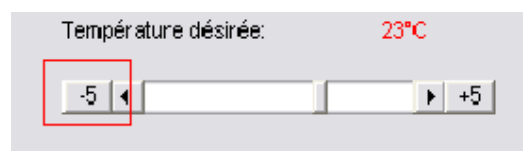


Figure 30: Le bouton -5

Le code :

```
Private Sub cmdPlus5_Click()  
    ' Quand on appuie ce bouton on diminue la consigne avec 5  
    intSetpoint = intSetpoint - 5  
    ' Pour mettre la limite inférieure (-10°C) à la consigne:  
    If intSetpoint > -10 Then  
        intSetpoint = -10  
    End If  
    hscSetpoint.Value = intSetpoint  
End Sub
```

Le scroll-bar:



Figure 31: scroll-bar pour le température

Le code:

```
Private Sub hscSetpoint_Change()  
    ' Quand on change la consigne avec le scroll-bar, on vient dans le routine _Change()  
    ' Quand le température est plus que 0°C, le text montre en rouge, sinon en bleu:  
    If hscSetpoint.Value > 0 Then  
        lblSetpoint2.ForeColor = &HFF&  
    Else  
        lblSetpoint2.ForeColor = &HC00000  
    End If  
    ' Pour mettre en mémoire la consigne avec la valeur du scroll-bar  
    intSetpoint = hscSetpoint.Value  
    lblSetpoint2.Caption = hscSetpoint.Value & "°C"  
End Sub
```

3.2.4 Bouton Exit



Figure 32: Le bouton "exit"

Le code :

```
Private Sub cmdSluiten_Click()  
' Appuyer sur ce bouton et le programme ferme  
End  
End Sub
```

3.2.5 Choix de la force du vent

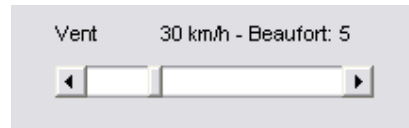


Figure 33: scroll-bar pour la force du vent

Le code :

```
Private Sub hscWind_Change()  
' Avec une structure de "Case" montrer le force de la vent en l'échelle du Beaufort  
Select Case hscWind.Value  
    Case Is <= 1  
        strBeaufort = "0"  
    Case 1 To 11  
        strBeaufort = "1-2"  
    Case 12 To 28  
        strBeaufort = "3-4"  
    Case 29 To 38  
        strBeaufort = "5"  
    Case 39 To 49  
        strBeaufort = "6"  
    Case 50 To 61  
        strBeaufort = "7"  
    Case 62 To 74  
        strBeaufort = "8"  
    Case 75 To 88  
        strBeaufort = "9"  
    Case 89 To 102  
        strBeaufort = "10"  
    Case 103 To 117  
        strBeaufort = "11"  
    Case Is > 117  
        strBeaufort = "12-14"  
    Case Else  
End Select  
  
lblWindtext.Caption = hscWind.Value & " km/h " & "- Beaufort: " & strBeaufort  
  
If optDuister.Value = True Then  
' Quand il fait nuit les stores doivent descendre quand le force du vent n'est pas trop  
' élevé et les lumières doivent être allumées  
  
If hscWind.Value > 70 Then  
    ' Le force du vent est plus que 70 km/h, puis les stores doivent rester en haut,  
    ' sinon le vent peut les détruire  
    img3d.Picture = LoadPicture(App.Path & "\licht_op.bmp")
```

```

Else
    'Le force du vent est moins que 70 km/h, les stores peuvent descendre
    img3d.Picture = LoadPicture(App.Path & "\licht_af.bmp")
End If

Else
    ' Quand il fait jour, les stores doivent rester en haut
    ' et les lumières ne peuvent pas être allumées
    img3d.Picture = LoadPicture(App.Path & "\standaard.bmp")
End If

End Sub

```

3.2.6 Choix de la clarté

Quand on choisit pour “Jour”, les lumières sont éteintes.



Figure 34: choix pour la clarté

Le code :

```

Private Sub optLicht_Click()
    ' Quand on choisit pour l'option "Jour", les lumières sont éteintes.
    img3d.Picture = LoadPicture(App.Path & "\standaard.bmp")
End Sub

```

Quand on choisit pour “Nuit”, les lumières sont allumées et les stores sont en haut ou en bas, suivant la force du vent.



Figure 35: choix pour la nuit

Le code:

```
Private Sub optDuister_Click()  
' Quand on choisit pour "Nuit", les lumières sont allumées  
' et les stores sont en haut ou en bas, dépendant de la force du vent  
If hscWind.Value > 50 Then  
' Stores en haut: Force du vent > 70km/u)  
    img3d.Picture = LoadPicture(App.Path & "\licht_op.bmp")  
Else  
' Stores en bas: Force du vent < 50km/u)  
    img3d.Picture = LoadPicture(App.Path & "\licht_af.bmp")  
End If  
End Sub
```

3.2.7 Choix du régulateur

Quand on choisit pour le régulateur PID,

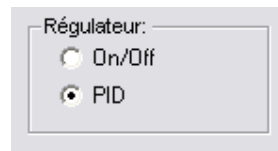


Figure 36: choix pour PID

le programme doit montrer les “TextBox” pour ajuster les variables.

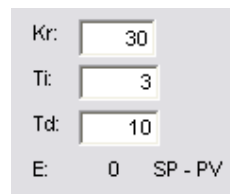


Figure 37: variables pour le PID

Le code :

```
Private Sub optPid_Click()  
' Quand on choisit pour le régulateur PID,  
' le programme doit montrer les "TextBox" pour ajuster les variables.  
' ça on fait avec l'option .Visible  
lblTd.Visible = True  
txtTd.Visible = True  
lblTi.Visible = True  
txtTi.Visible = True  
lblKr.Visible = True  
txtKr.Visible = True  
lblHysteresis.Visible = False  
txtHysteresis.Visible = False  
End Sub
```

Quand on choisit le régulateur On/Off,



Figure 38: choix pour le On/Off

le programme doit montrer les “TextBox” pour ajuster la valeur d’hystérésis.

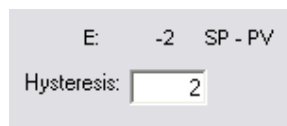


Figure 39: hystérèse

Le code :

```
Private Sub optOnOff_Click()
' Quand on choisit pour le régulateur On/Off,
' le programme doit montrer les "TextBox" pour ajuster le hystérèse.
' ça on fait avec l'option .Visible
lblTd.Visible = False
txtTd.Visible = False
lblTi.Visible = False
txtTi.Visible = False
lblKr.Visible = False
txtKr.Visible = False
lblHysteresis.Visible = True
txtHysteresis.Visible = True
End Sub
```

3.2.8 Le Timer

On a besoin d’un « Timer » parce le programme doit être dans une boucle et rester actif jusqu’à l’utilisateur ferme le programme.

```
Private Sub Timer1_Timer()
' Definition de la variable "intVerandering"
intVerandering = (intVerandering * 30) / 60
```

```
On Error Resume Next
```

```
' Quand les valeurs de txtKr, txtTi, ou txtTd sont plus que 100,
' elles sont ramenées vers 100 et quand elles sont moins que 0,
```

```

' elles sont ramenées vers 0. Le hystérèse devient aussi limiter
' entre 1 et 7.
If txtKr < 0 Then
    txtKr = 0
End If
If txtKr > 100 Then
    txtKr = 100
End If
dblKr = Val(txtKr)

If txtTi < 0 Then
    txtTi = 0
End If
If txtTi > 100 Then
    txtTi = 100
End If
dblTi = Val(txtTi)

If txtTd < 0 Then
    txtTd = 0
End If
If txtTd > 100 Then
    txtTd = 100
End If
dblTd = Val(txtTd)

If txtHysteresis < 1 Then
    txtHysteresis = 1
End If
If txtHysteresis > 7 Then
    txtHysteresis = 7
End If
' -----
' Calcul de l'erreur, la différence entre la température mesurée et la consigne.
' Montrer l'erreur et la température mesurée
lblPv.Caption = CInt(dblProceswaarde) & "°C"
dblError = intSetpoint - dblProceswaarde
lblError.Caption = CInt(dblError)

' Code pour appeler la routine du régulateur PID
If optPid.Value = True Then
    ' Code pour visualiser la température mesurée quand il tombe dehors la graphique.
    If dblProceswaarde < 50 Then
        dblProceswaarde = dblProceswaarde + intToevoer
    End If
    If dblProceswaarde > -10 Then
        dblProceswaarde = dblProceswaarde - intVerandering
    End If
pid
End If

```



```

' Code pour appeler la routine du régulateur On/Off
If optOnOff.Value = True Then
    aan_uit
End If

' Code pour la graphique et pour montrer la température mesurée
pctGrafiek.Cls
intPvgrafiek(100) = dblProceswaarde
For intX = 0 To 99
    intPvgrafiek(intX) = intPvgrafiek(intX + 1)
    pctGrafiek.PSet (intX, 70 - (intPvgrafiek(intX)))
Next intX

' Code pour montrer la consigne en la graphique
' Avec .Line on dessine une ligne (en jaune -> vbYellow)
pctGrafiek.Line (0, 70 - intSetpoint)-(100, 70 - intSetpoint), vbYellow
End Sub

```

3.2.9 Le régulateur PID

Le code du régulateur PID :

```

Private Sub pid()
' On a besoin d'un filtre pour limiter l'action-D
intFilter = 10
' L'input du régulateur est la température mesurée + la différence avec
' la dernière température mesurée multiplier avec un facteur qui consiste
' le variable Td
dblInputD = dblProceswaarde + (dblVorige - dblProceswaarde) * (dblTd / 60)
' Pour mémoriser la dernière température mesurée
dblVorige = dblProceswaarde

dblInputDF = dblInputDF + (dblInputD - dblInputDF) * intFilter / 60
dblOutput = (intSetpoint - dblInputDF) * (dblKr / 100) + dblFeedback
' Limiter l'output entre 0 et 100%
If dblOutput > 100 Then
    dblOutput = 100
End If
If dblOutput < 0 Then
    dblOutput = 0
End If
' Donner le changement
intVerandering = 100 - dblOutput
dblFeedback = dblFeedback - (dblFeedback - dblOutput) * dblTi / 60

' -----
' Pour montrer les images correctes on divise cette partie du programme en deux:
' Une partie quand il fait nuit, et l'autre partie quand il fait jour
' Quand il fait nuit (optDuister.Value=true):

```

```

If optDuister.Value = True Then

    If dblProceswaarde > intSetpoint Then
        ' la température mesurée est plus que la consigne
        imgDoorsnede.Picture = LoadPicture(App.Path & "\verwarming_min_licht_aan.bmp")
    End If

    If dblProceswaarde < intSetpoint Then
        ' la température mesurée est moins que la consigne
        imgDoorsnede.Picture = LoadPicture(App.Path & "\verwarming_veel_licht_aan.bmp")
    End If

    If dblProceswaarde = intSetpoint Then
        ' la température mesurée est égale à la consigne
        imgDoorsnede.Picture = LoadPicture(App.Path & "\verwarming_aan_licht_aan.bmp")
    End If

End If

' -----
' Quand il fait jour (optLicht.Value=true):
If optLicht.Value = True Then

    If dblProceswaarde > intSetpoint Then
        ' la température mesurée est plus que la consigne
        imgDoorsnede.Picture = LoadPicture(App.Path & "\verwarming_min_licht_uit.bmp")
    End If

    If dblProceswaarde < intSetpoint Then
        ' la température mesurée est moins que la consigne
        imgDoorsnede.Picture = LoadPicture(App.Path & "\verwarming_veel_licht_uit.bmp")
    End If

    If dblProceswaarde = intSetpoint Then
        ' la température mesurée est égale à la consigne
        imgDoorsnede.Picture = LoadPicture(App.Path & "\verwarming_aan_licht_uit.bmp")
    End If

End If

End Sub

```

3.2.10 Le régulateur On/Off

Le code du régulateur On/Off :

```

Private Sub aan_uit()
intSetpoint = hscSetpoint.Value
intHysteresis = Val(txtHysteresis)
' On doit établir une limite supérieure et inférieure pour la bande d'hystérésis

```

```

' "intOnder" est la limite inférieure
' "intBoven" est la limite supérieure
intOnder = intSetpoint - intHysteresis
intBoven = intHysteresis + intSetpoint

' Pour montrer les images correctes on divise cette partie du programme en deux:
' Une partie quand il fait nuit, et l'autre partie quand il fait jour
' Quand il fait jour (optLicht.Value=true):
If optLicht.Value = True Then

    If (dblProceswaarde < intOnder) Or (dblProceswaarde > intBoven) Then
        ' Quand la température mesurée est moins que la limite inférieure ou plus que
        ' la limite supérieure.
        If dblProceswaarde >= intBoven Then
            ' Quand la température mesurée est plus que ou égale à la limite supérieure,
            ' on doit diminuer l'output
            dblProceswaarde = dblProceswaarde - 1
            imgDoorsnede.Picture = LoadPicture(App.Path & "\verwarming_uit_licht_uit.bmp")
        End If

        If dblProceswaarde <= intOnder Then
            ' Quand la température mesurée est plus que ou égal à la limite supérieure,
            ' on doit augmenter l'output
            dblProceswaarde = dblProceswaarde + 1
            imgDoorsnede.Picture = LoadPicture(App.Path & "\verwarming_aan_licht_uit.bmp")
        End If
    Else
        ' Quand la température mesurée est entre la limite inférieure et la limite supérieure.
        If dblProceswaarde = (intOnder) Then
            ' Quand la température mesurée est égale à la limite inférieure.
            intA = 1
        End If

        If dblProceswaarde = (intBoven) Then
            ' Quand la température mesurée est égale à la limite supérieure.
            intA = 2
        End If

        Select Case intA
            Case 1
                ' Quand la température mesurée est égale à la limite inférieure, on augmente l'output,
                ' jusqu'à la température mesurée est égale à la limite supérieure
                dblProceswaarde = dblProceswaarde + 1
                imgDoorsnede.Picture = LoadPicture(App.Path & "\verwarming_aan_licht_uit.bmp")
            Case 2
                ' Quand la température mesurée est égale à la limite supérieure on diminue l'output,
                ' jusqu'à la température mesurée est égale à la limite inférieure
                dblProceswaarde = dblProceswaarde - 1
                imgDoorsnede.Picture = LoadPicture(App.Path & "\verwarming_uit_licht_uit.bmp")
        End Select
    End If
End If

```

End If
End If

```
'-----  
' Quand il fait nuit (optDuister.Value=true):  
If optDuister.Value = True Then  
  
    If (dblProceswaarde < intOnder) Or (dblProceswaarde > intBoven) Then  
        ' Quand la température mesurée est moins que la limite inférieure ou plus que  
        ' la limite supérieure.  
        If dblProceswaarde >= intBoven Then  
            ' Quand la température mesurée est plus que ou égale à la limite supérieure,  
            ' on doit diminuer l'output  
            dblProceswaarde = dblProceswaarde - 1  
            imgDoorsnede.Picture = LoadPicture(App.Path & "\\verwarming_uit_licht_aan.bmp")  
        End If  
  
        If dblProceswaarde <= intOnder Then  
            ' Quand la température mesurée est plus que ou égal à la limite supérieure,  
            ' on doit augmenter l'output  
            dblProceswaarde = dblProceswaarde + 1  
            imgDoorsnede.Picture = LoadPicture(App.Path & "\\verwarming_aan_licht_aan.bmp")  
        End If  
    Else  
        ' Quand la température mesurée est entre la limite inférieure et la limite supérieure.  
  
        If dblProceswaarde = (intOnder) Then  
            ' Quand la température mesurée est égale à la limite inférieure.  
            intA = 1  
        End If  
  
        If dblProceswaarde = (intBoven) Then  
            ' Quand la température mesurée est égale à la limite supérieure.  
            intA = 2  
        End If  
  
        Select Case intA  
            Case 1  
                ' Quand la température mesurée est égale à la limite inférieure, on augmente l'output,  
                ' jusqu' à la température mesurée est égale à la limite supérieure  
                dblProceswaarde = dblProceswaarde + 1  
                imgDoorsnede.Picture = LoadPicture(App.Path &  
                "\\verwarming_aan_licht_aan.bmp")  
            Case 2  
                ' Quand la température mesurée est égale à la limite supérieure on diminue l'output,  
                ' jusqu' à la température mesurée est égale à la limite inférieure  
                dblProceswaarde = dblProceswaarde - 1  
                imgDoorsnede.Picture = LoadPicture(App.Path & "\\verwarming_uit_licht_aan.bmp")  
        End Select  
    End If
```

End If
End Sub

3.2.11 **Module avec les variables**

Le module est un autre fichier où les variables y sont stockés.

Le code :

```
Option Explicit
' les variables définits comme "integer":
' le nombre entier, 2 bytes
Global intWaarde As Integer
Global intToevoer As Integer
Global intVerandering As Integer
Global intSetpoint As Integer
Global intTaal As Integer
Global intPvgrafiek(100) As Integer
Global intFilter As Integer
Global intX As Integer
Global intHysteresis As Integer
Global intOnder As Integer
Global intBoven As Integer
Global intA As Integer

' les variables définits comme "double":
' virgule flottante, 8 bytes
Global dblProceswaarde As Double
Global dblKr As Double
Global dblTi As Double
Global dblTd As Double
Global dblError As Double
Global dblFeedback As Double
Global dblVorige As Double
Global dblOutput As Double
Global dblInputD As Double
Global dblInputDF As Double
Global dblInput As Double

' les variables définits comme "string":
' texte
Global strBeaufort As String
```

4 Régulateurs

On a choisi pour les régulateurs On/Off et PID, parce que le premier régulateur est le régulateur conventionnel type thermostat. Et le choix pour le régulateur PID, c'est parce qu'il donne le meilleur résultat.

4.1 Régulateur On/Off avec hystérésis

Un régulateur on/off est la forme la plus simple d'un régulateur. L'output est allumé ou l'output est éteint, sans état moyen. Par exemple, un thermostat est un régulateur simple de « négatif-rétroaction » : quand la température (la valeur mesurée) va au-dessous d'un point de réglage, le réchauffeur est allumé. Autre exemple est peut-être un régulateur de pression sur un compresseur d'air : quand la pression (la valeur mesurée) chute au-dessous de la valeur désirée, la pompe est actionnée.

Cette manière de régler est valable mais des éléments mouvants, comme les soupapes ou les clapets meurent avec une fréquence très grande et ils s'usent très vite.

C'est pourquoi on utilise souvent un régulateur on/off avec hystérésis. Une région autour de la valeur désirée, où aucune régulation ne se produit. Seulement quand la valeur du système atteint la limite supérieure ou la limite inférieure, le régulateur commence à marcher.

D'abord, quand la température est hors la bande d'hystérésis, le régulateur s'occupe d'emmener la température entre la limite supérieure et la limite inférieure en allumant ou éteignant l'output. Puis, quand la température est entre ces deux limites. Le procédé réagit comme sur graphique ci-dessous.

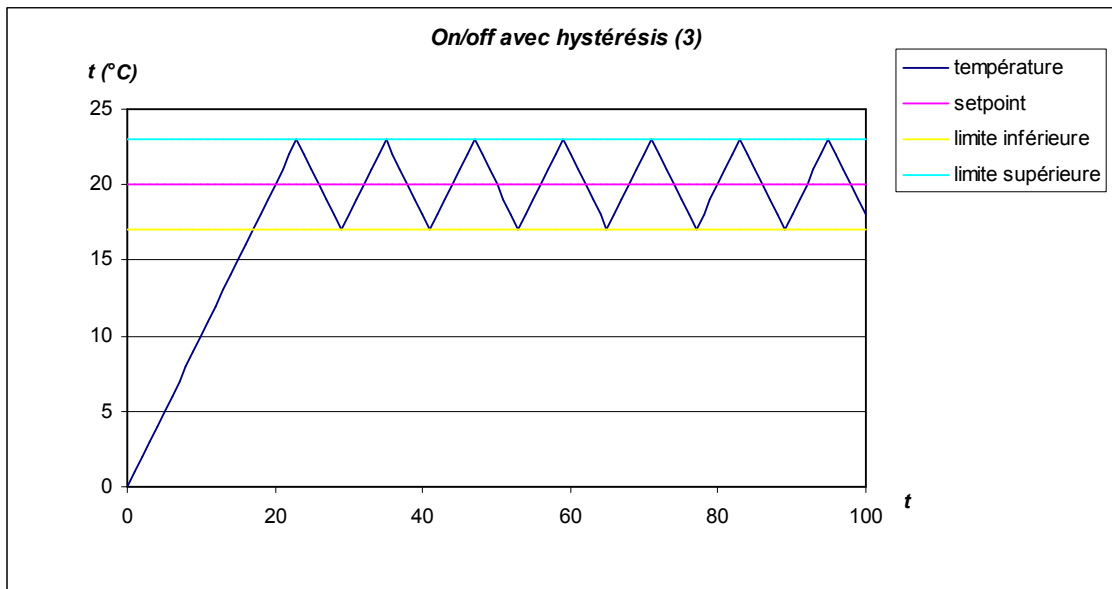


Figure 40 : Graphique du régulateur On/Off

Pseudo code pour programmer un régulateur On/Off:

```
If valeur_mésurée ≤ limite_inférieur Or valeur_mésurée ≥ limite_supérieur Then
  If valeur_mésurée > limite_supérieur Then
    valeur_mésurée - x
  End If

  If valeur_mésurée < limite_inférieur Then
    valeur_mésurée + x
  End If
End If

If valeur_mésurée = limite_inférieur Then
  valeur_mésurée + x
End If

If valeur_mésurée = limite_supérieur Then
  valeur_mésurée -x
End If
```


4.2 Régulateur PID

Le régulateur PID fonctionne avec la différence entre les valeurs mesurées et les valeurs désirées. Ce type de régulateur est disposé de trois actions différentes.

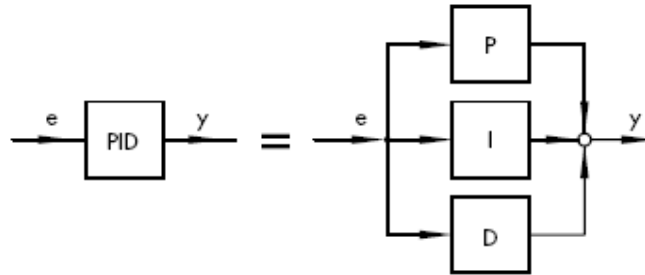


Figure 41: Principe du régulateur PID

Quand on place le régulateur dans le boucle de régulation, le processus est le suivant :

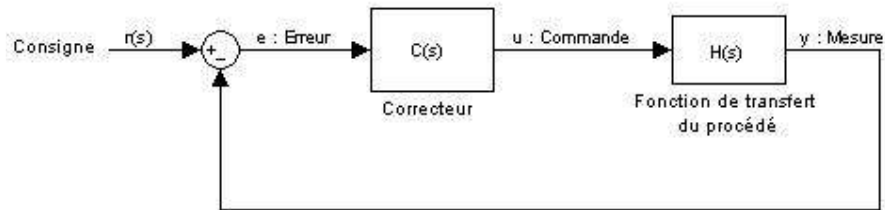


Figure 42: boucle de régulation

4.2.1 Action proportionnelle

L'action P (proportionnelle) : avec l'action proportionnelle, le signal de terminaison est directement proportionnel avec la hauteur du signal d'entrée. Au signal de terminaison, un renforcement intervient. Lorsque K_r augmente, le temps de montée est plus court mais il y a un dépassement plus important. Le temps d'établissement varie peu et l'erreur statique se trouve améliorée.

Dans la figure ci-dessous, vous voyez le principe fonctionnement de l'action proportionnelle.

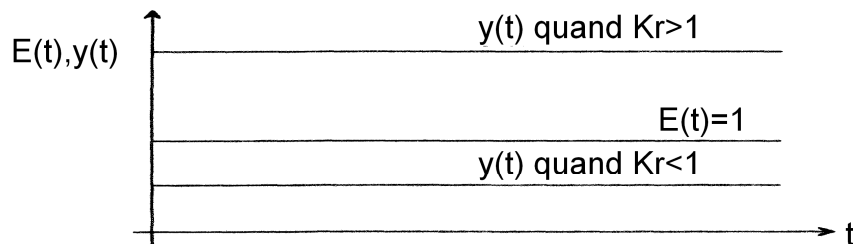


Figure 43: réponse échelon du régulateur P

On doit chercher un compromis entre stabilité et rapidité. Parce que quand on choisit le K_r trop élevé, le procédé devient instable.

Dans la figure suivante on peut voir le « root-locus » d'un procédé. Quand les racines sont dans le domaine gauche, le système est stable.

Le « root-locus » est un outil pour analyser les systèmes dynamiques et continus avec un input et un output (single input single output : SISO). « Root-locus » donne le lieu des racines d'une fonction de transfert dépendant de la fortification (K_r). Le graphique du « root-locus » se trouve dans le plan Laplace.

Ci-dessous on voit que les racines (en rouge) sont dans le domaine de gauche, le domaine stable.

Par exemple pour la fonction de transfert :
$$G(s) = \frac{2}{(7s + 1)(3s + 1)(2s + 1)(s + 1)}$$

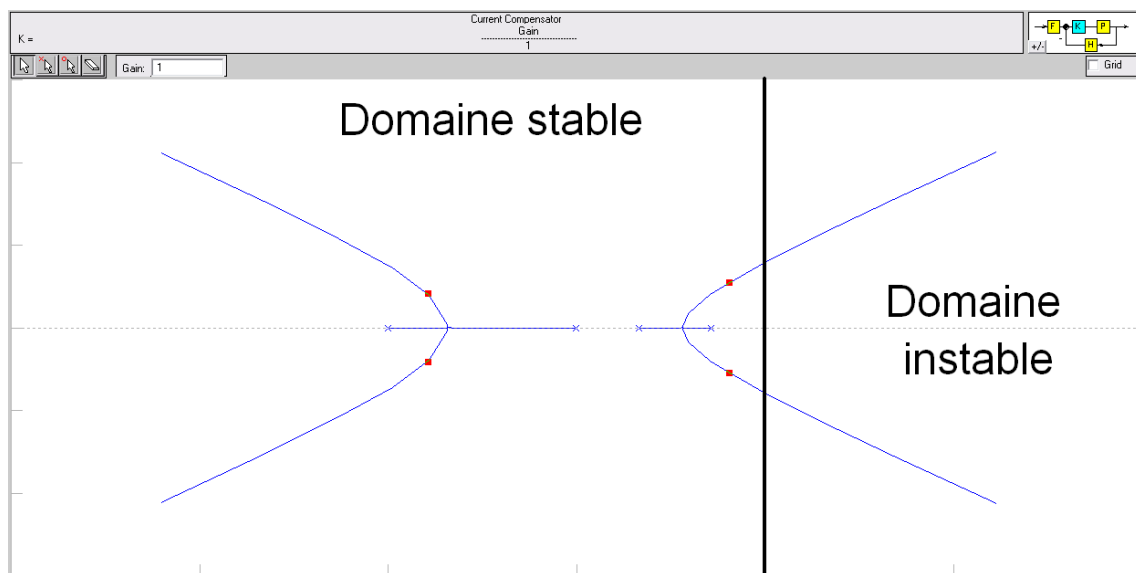


Figure 44: root-locus d'un procédé stable

Dans cette réponse échelon, on voit que le procédé est stable, elle continue à une valeur constante. Avec la réponse échelon, on donne un changement à l'input et on voit comment le output réagit. Avec ce graphique on est en état d'analyser le procédé et former le fonction de transfert avec les méthodes de Strejc, Broida, Van Der Grinten, Sundaresan,...

La réponse échelon est le test le plus utiliser, avec ce test on peut voir si le procédé est stable, quand il continue vers une valeur constante, ou instable, quand il commence à osciller.

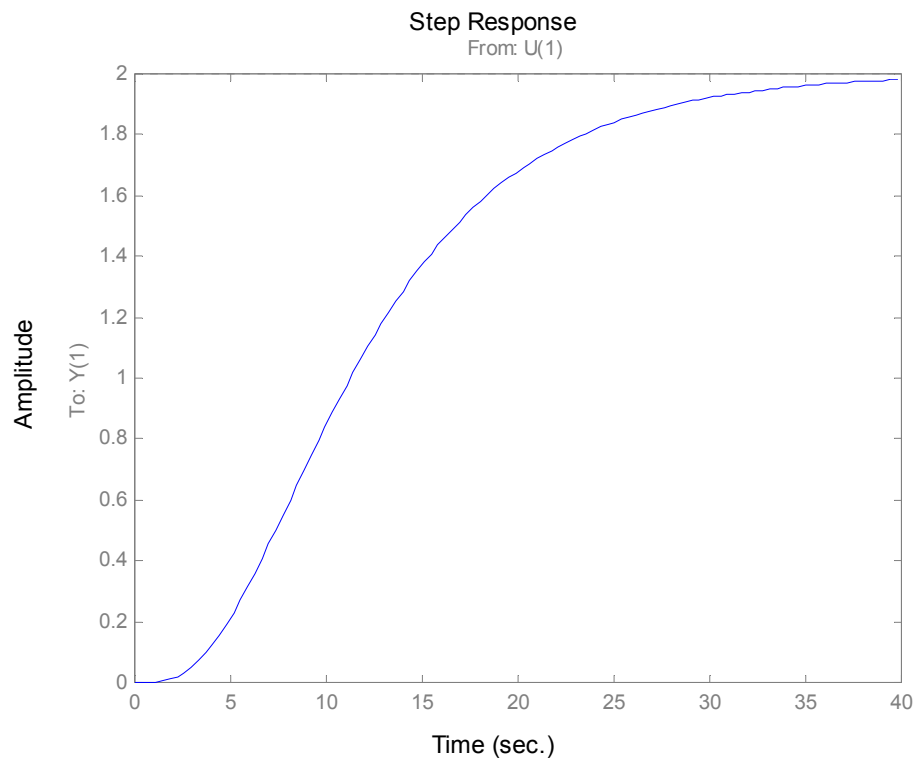


Figure 45: réponse échelon d'un procédé stable

Dans cette figure on voit que deux racines sont dans le domaine droit, le domaine instable

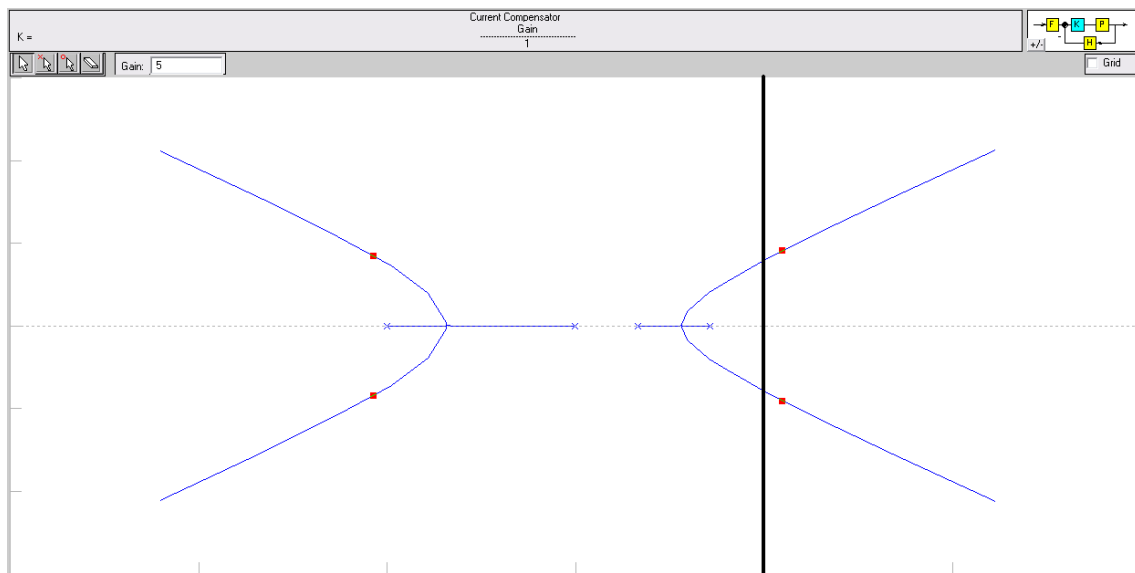


Figure 46: root-locus d'un procédé instable

Quand on voit la figure suivante, le procédé commence à osciller et ce n'est plus contrôlable.

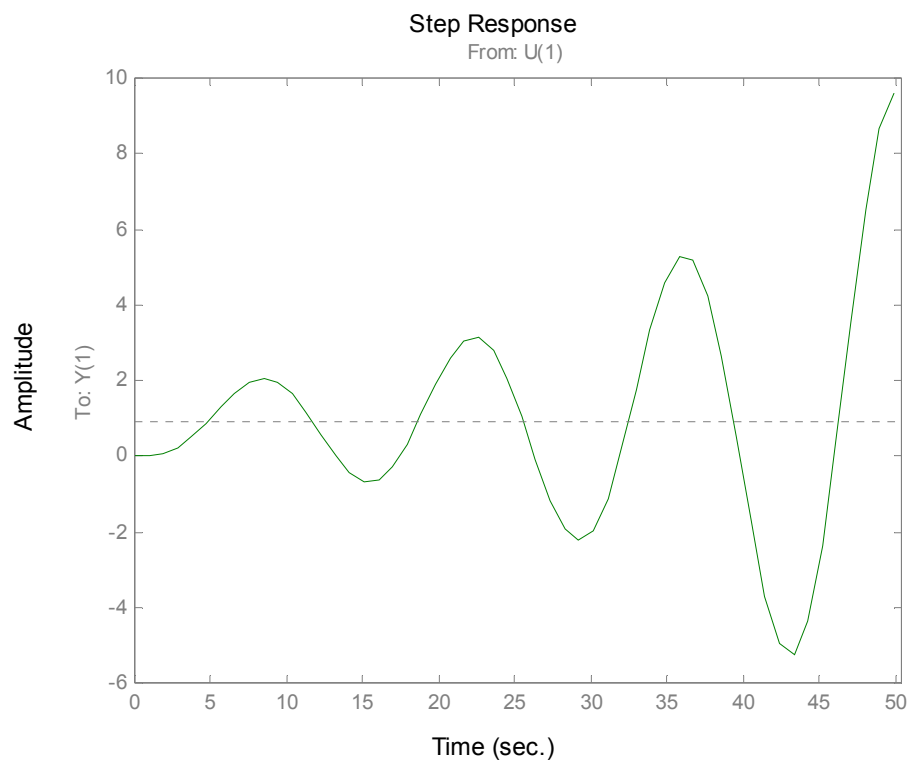


Figure 47: réponse échelon d'un procédé instable

Puis pour chercher la limite de la fortification on utilise l'array de Routh-Hurwitz

En premier on doit chercher la fonction de transfert de la boucle fermée:

Le procédé est par exemple (boucle ouverte): $G(s) = \frac{2}{(7s+1)(3s+1)(2s+1)(s+1)}$

Puis la fonction pour laisser varier la fortification : $R(s) = Kr$

Ça on donne en boucle ouverte : $G(s) \times R(s) = \frac{2Kr}{42s^4 + 83s^3 + 53s^2 + 13s + 1}$

Pour fermer la boucle : $G(s) \times R(s) = \frac{Te(s)}{No(s)} \Rightarrow T(s) = \frac{Te(s)}{Te(s) + No(s)}$

$$T(s) = \frac{2Kr}{42s^4 + 83s^3 + 53s^2 + 13s + (1 + 2Kr)}$$

La stabilité est seulement dépendante du dénominateur parce que, seulement les racines et pas les zéros (du numérateur) déterminent la stabilité. Ensuite on a l'équation caractéristique :

$$42s^4 + 83s^3 + 53s^2 + 13s + (1 + 2Kr)$$

Maintenant on est en état de former l'array de Routh-Hurwitz :

$$\begin{array}{cccc} 42 & 53 & 1+2Kr & 0 \\ 83 & 13 & 0 & 0 \\ A1 & A2 & 0 & 0 \\ B1 & B2 & 0 & 0 \\ C1 & 0 & 0 & 0 \\ D1 & 0 & 0 & 0 \end{array}$$

D'abord on calcule les facteurs A1 et A2, et on les remplit dans l'array :

$$A1 = -\frac{\begin{vmatrix} 42 & 53 \\ 83 & 13 \end{vmatrix}}{83} = -\frac{42 \times 13 - 83 \times 53}{83} = 46,42$$

$$A2 = -\frac{\begin{vmatrix} 42 & 1+2Kr \\ 83 & 0 \end{vmatrix}}{83} = -\frac{42 \times 0 - 83 \times (1 + 2Kr)}{83} = 1 + 2Kr$$

$$\begin{vmatrix} 42 & 53 & 1+2Kr & 0 \\ 83 & 13 & 0 & 0 \\ 46,42 & 1+2Kr & 0 & 0 \\ B1 & B2 & 0 & 0 \\ C1 & 0 & 0 & 0 \\ D1 & 0 & 0 & 0 \end{vmatrix}$$

Puis les facteurs B1 et B2 et les remplir :

$$B1 = -\frac{\begin{vmatrix} 42 & 13 \\ 46,42 & 1+2Kr \end{vmatrix}}{46,42} = -\frac{83 \times (1+2Kr) - 46,42 \times 13}{46,42} = \frac{520,46 - 166Kr}{46,42}$$

$$B2 = -\frac{\begin{vmatrix} 83 & 0 \\ 46,42 & 0 \end{vmatrix}}{46,42} = -\frac{83 \times 0 - 46,42 \times 0}{46,42} = 0$$

$$\begin{vmatrix} 42 & 53 & 1+2Kr & 0 \\ 83 & 13 & 0 & 0 \\ 46,42 & 1+2Kr & 0 & 0 \\ \frac{520,46 - 166Kr}{46,42} & 0 & 0 & 0 \\ C1 & 0 & 0 & 0 \\ D1 & 0 & 0 & 0 \end{vmatrix}$$

Puis le facteur C1 et le remplir :

$$C1 = -\frac{\begin{vmatrix} 46,42 & 1+2Kr \\ \frac{520,46 - 166Kr}{46,42} & 0 \end{vmatrix}}{\frac{520,46 - 166Kr}{46,42}} = -\frac{46,42 \times 0 - \frac{520,46 - 166Kr}{46,42} \times (1+2Kr)}{\frac{520,46 - 166Kr}{46,42}} = 1+2Kr$$

$$\begin{array}{cccc|c}
 42 & 53 & 1+2Kr & 0 & \\
 83 & 13 & 0 & 0 & \\
 46,42 & 1+2Kr & 0 & 0 & \\
 \hline
 520,46-166Kr & 0 & 0 & 0 & \\
 46,42 & & & & \\
 1+2Kr & 0 & 0 & 0 & \\
 D1 & 0 & 0 & 0 &
 \end{array}$$

Puis le facteur D1 et le remplir :

$$D1 = - \frac{\begin{vmatrix} 520,46-166Kr & 0 \\ 46,42 & 0 \end{vmatrix}}{1+2Kr} = - \frac{\frac{520,46-166Kr}{46,42} \times 0 - (1+2Kr) \times 0}{1+2Kr} = 0$$

Comme array final on a le suivant :

$$\begin{array}{cccc|c}
 42 & 53 & 1+2Kr & 0 & \\
 83 & 13 & 0 & 0 & \\
 46,42 & 1+2Kr & 0 & 0 & \\
 \hline
 520,46-166Kr & 0 & 0 & 0 & \\
 46,42 & & & & \\
 1+2Kr & 0 & 0 & 0 & \\
 0 & 0 & 0 & 0 &
 \end{array}$$

Toutes les valeurs de la première colonne doivent être positives. On sait aussi que Kr ne peut pas être négatif, puis on trouve l'équation suivante :

$$\frac{520,46-166Kr}{46,42} \geq 0 \Rightarrow k \leq 3,13$$

Quand on utilise cette fortification le graphique de « root-locus » est comme le suivant :

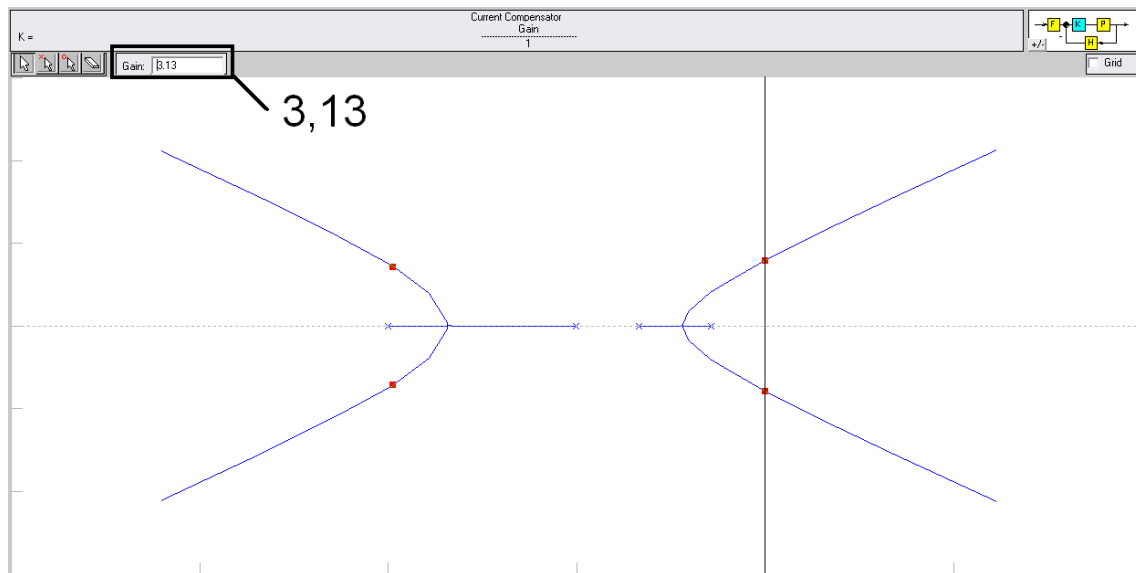


Figure 48: root-locus d'un procédé avec fortification critique

Quand on fait la réponse échelon, ça donne la suivante :

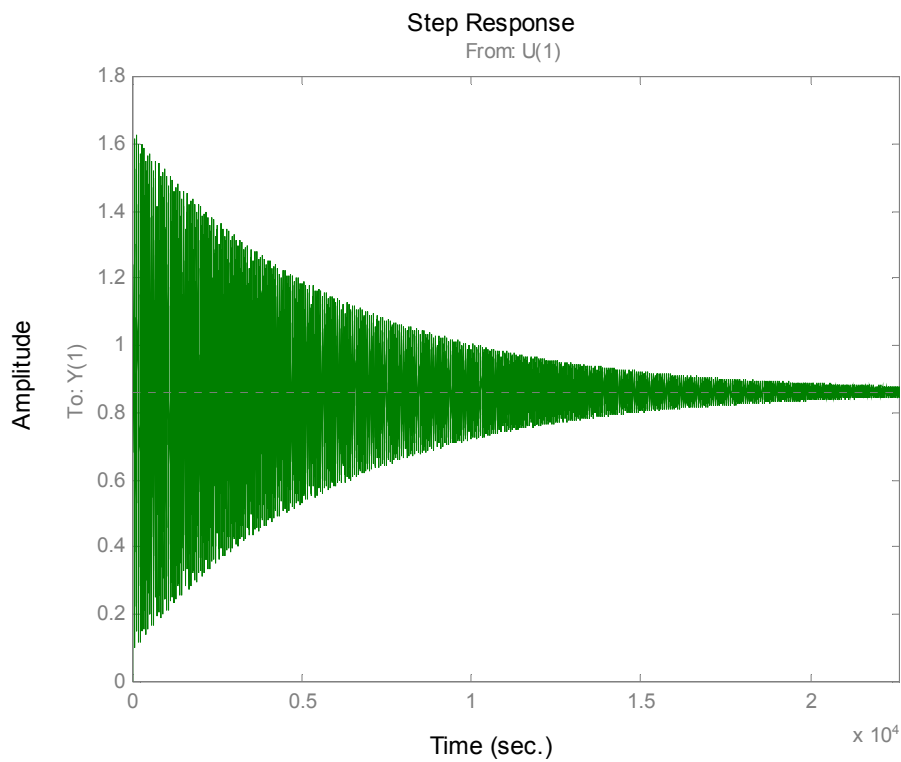


Figure 49: réponse échelon d'un procédé avec fortification critique

On voit que au début il y a une oscillation forte, mais après le procédé prend une valeur constante.

4.2.2 Action intégrante

L'action I (intégrante) : l'action intégrante, le signal de terminaison continue à augmenter aussi si long qu'un signal d'entrée positif est présent. Si il n'y a aucun signal d'entrée, le signal de terminaison restera inchangé et si le signal d'entrée est négatif, le signal de terminaison diminuera. Lorsque $1/T_i$ augmente, le temps de montée est plus court mais il y a un dépassement plus important. Le temps d'établissement au régime stationnaire s'allonge mais dans ce cas on assure une erreur statique nulle.

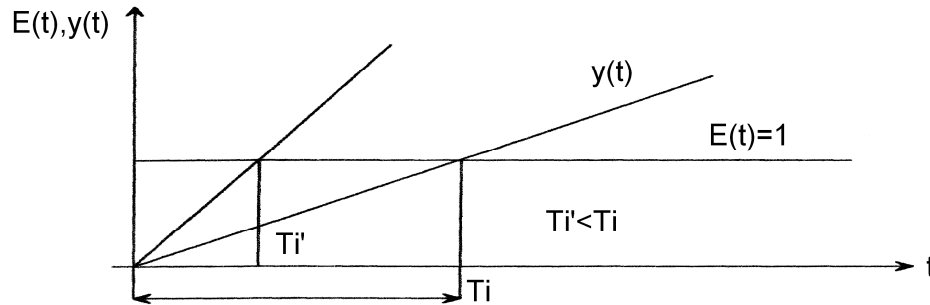


Figure 50: réponse échelon du régulateur I

4.2.3 Action différentielle

L'action D (différentielle) : avec l'action différentielle la hauteur du signal de terminaison est dépendant du changement du signal d'entrée. Quand le signal d'entrée est constant il n'y a aucun signal de terminaison. Avec cette action, un changement rapide est créé dans le début. Lorsque T_d augmente, le temps de montée change peu mais le dépassement diminue. Le temps d'établissement au régime stationnaire est meilleur. L'action différentielle réagit plus rapidement, mais malheureusement elle réagit aussi au bruit léger des mesures. C'est pourquoi $T_d \ll T_i$. L'action différentielle ne cause pas d'influences sur l'erreur statique. On ne peut pas utiliser un régulateur D, parce que il réagit seulement au changement, puis on utilise un régulateur PD.

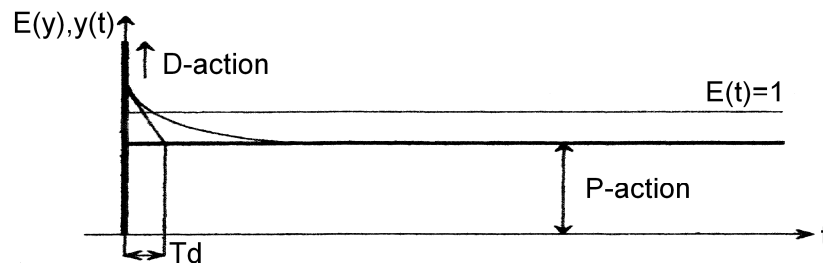


Figure 51: réponse échelon du régulateur PD

4.2.4 Combinaison : PID

La combinaison de ces trois actions mentionnées ci-dessus, est intégré d'un régulateur PID. Dans le signal de terminaison du PID, un couple des caractéristiques est séparé. L'action proportionnelle s'occupe de la fonction d'étape. La fonction intégrante cause un signal ce qui augmente de façon linéaire avec le temps. L'action différentielle s'occupe du voltage de pic dans le début.

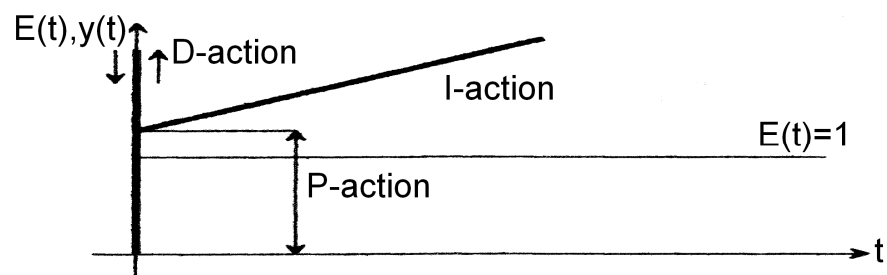


Figure 52: réponse échelon théorique du PID

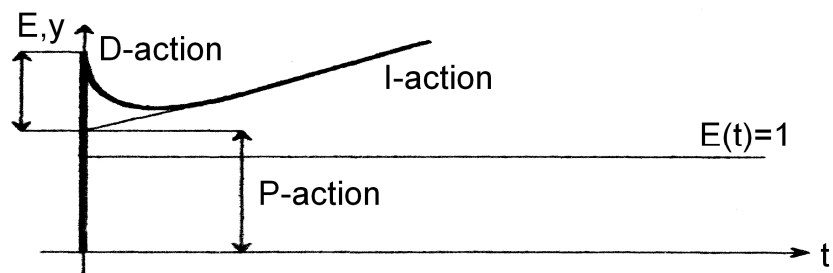


Figure 53: réponse échelon pratique du PID

5 Bibliographie

FRANS, R. *Leren programmeren met Visual Basic 6.0*, Campania Media, 2001.

GOFFIN, W. / MEURIS, F. *Cursus Regeltechniek*, KHLim, 1994.

DANIELS C. *Cursus Visual Basic*, KHLim.

GOFFIN W. *Regeltechniek*, KHLim.

AUTODESK *Tutorials* (<http://students5.autodesk.com>), 2007.

6 Annexe 1: Le programme

frmTaal :

```
Private Sub formLoad()  
frmProject.Hide  
frmTaal.Show  
End Sub
```

```
Private Sub cmdDuits_Click()  
intTaal = 4  
frmTaal.Hide  
frmProject.Show  
End Sub
```

```
Private Sub cmdEngels_Click()  
intTaal = 1  
frmTaal.Hide  
frmProject.Show  
End Sub
```

```
Private Sub cmdFrans_Click()  
intTaal = 2  
frmTaal.Hide  
frmProject.Show  
End Sub
```

```
Private Sub cmdNederlands_Click()  
intTaal = 3  
frmTaal.Hide  
frmProject.Show  
End Sub
```

frmProject :

```
Option Explicit  
Private Sub cmdMin5_Click()  
intSetpoint = intSetpoint - 5  
If intSetpoint < -10 Then  
    intSetpoint = -10  
End If  
hscSetpoint.Value = intSetpoint  
End Sub
```

```
Private Sub cmdPlus5_Click()  
intSetpoint = intSetpoint + 5  
If intSetpoint > 40 Then  
    intSetpoint = 40
```

```
End If
hscSetpoint.Value = intSetpoint
End Sub
```

```
Private Sub cmdSluiten_Click()
End
End Sub
```

```
Private Sub Form_load()
frmProject.Left = (Screen.Width / 2) - (frmProject.Width / 2)
frmProject.Top = (Screen.Height / 2) - (frmProject.Height / 2)
```

```
imgDoorsnede.Picture = LoadPicture(App.Path & "\verwarming_aan_licht_uit.bmp")
img3d.Picture = LoadPicture(App.Path & "\standaard.bmp")
```

```
Select Case intTaal
```

```
Case 1
```

```
    lblWind.Caption = "Wind"
    lblSetpoint.Caption = "Desired temperature:"
    fraLicht.Caption = "Light:"
    optDuister.Caption = "Dark"
    optLicht.Caption = "Light"
    fraRegelaar.Caption = "Regulator:"
    lblTemp.Caption = "Current temperature:"
```

```
Case 2
```

```
    lblWind.Caption = "Vent"
    lblSetpoint.Caption = "Température désirée:"
    fraLicht.Caption = "Clarté:"
    optDuister.Caption = "Nuit"
    optLicht.Caption = "Jour"
    fraRegelaar.Caption = "Régulateur:"
    lblTemp.Caption = "Température présente:"
```

```
Case 3
```

```
    lblWind.Caption = "Wind"
    lblSetpoint.Caption = "Gewenste temperatuur:"
    fraLicht.Caption = "Lichtsterkte:"
    optDuister.Caption = "Donker"
    optLicht.Caption = "Licht"
    fraRegelaar.Caption = "Regelaar:"
    lblTemp.Caption = "Huidige temperatuur:"
```

```
Case 4
```

```
    lblWind.Caption = "Wind"
    lblSetpoint.Caption = "Gewünschte Temperatur:"
    fraLicht.Caption = "Licht:"
    optDuister.Caption = "Dunkel"
    optLicht.Caption = "Licht"
    fraRegelaar.Caption = "Regler"
```

```

        lblTemp.Caption = "Heutige Temperatur:"

End Select

hscSetpoint.Value = 23

intSetpoint = hscSetpoint.Value

lblPv.Caption = CInt(dblProceswaarde) & "°C"

intWaarde = 2000
intToevoer = (10 * (intWaarde / 100)) / 6

optPid.Value = True
optLicht.Value = True

dblKr = txtKr.Text
dblTi = txtTi.Text
dblTd = txtTd.Text

If optPid.Value = True Then
    lblTd.Visible = True
    txtTd.Visible = True
    lblTi.Visible = True
    txtTi.Visible = True
    lblKr.Visible = True
    txtKr.Visible = True
    lblHysteresis.Visible = False
    txtHysteresis.Visible = False
End If

pctGrafiek.Cls
pctGrafiek.ScaleMode = 3
pctGrafiek.ScaleHeight = 110
pctGrafiek.ScaleWidth = 100
pctGrafiek.AutoRedraw = True
pctGrafiek.ForeColor = vbCyan
pctGrafiek.DrawStyle = 0
pctGrafiek.DrawWidth = 1

End Sub

Private Sub hscWind_Change()
Select Case hscWind.Value
    Case Is <= 1
        strBeaufort = "0"
    Case 1 To 11
        strBeaufort = "1-2"
    Case 12 To 28
        strBeaufort = "3-4"

```

```

Case 29 To 38
    strBeaufort = "5"
Case 39 To 49
    strBeaufort = "6"
Case 50 To 61
    strBeaufort = "7"
Case 62 To 74
    strBeaufort = "8"
Case 75 To 88
    strBeaufort = "9"
Case 89 To 102
    strBeaufort = "10"
Case 103 To 117
    strBeaufort = "11"
Case Is > 117
    strBeaufort = "12-14"
Case Else
End Select

lblWindtext.Caption = hscWind.Value & " km/h " & "- Beaufort: " & strBeaufort

If optDuister.Value = True Then

    If hscWind.Value > 70 Then
        img3d.Picture = LoadPicture(App.Path & "\licht_op.bmp")

    Else
        img3d.Picture = LoadPicture(App.Path & "\licht_af.bmp")
    End If

Else
    img3d.Picture = LoadPicture(App.Path & "\standaard.bmp")
End If

End Sub

Private Sub optDuister_Click()
If hscWind.Value > 70 Then
    img3d.Picture = LoadPicture(App.Path & "\licht_op.bmp")
Else
    img3d.Picture = LoadPicture(App.Path & "\licht_af.bmp")
End If
End Sub

Private Sub optLicht_Click()
img3d.Picture = LoadPicture(App.Path & "\standaard.bmp")
End Sub

Private Sub optOnOff_Click()
lblTd.Visible = False

```

```
txtTd.Visible = False
lblTi.Visible = False
txtTi.Visible = False
lblKr.Visible = False
txtKr.Visible = False
lblHysteresis.Visible = True
txtHysteresis.Visible = True
End Sub
```

```
Private Sub optPid_Click()
lblTd.Visible = True
txtTd.Visible = True
lblTi.Visible = True
txtTi.Visible = True
lblKr.Visible = True
txtKr.Visible = True
lblHysteresis.Visible = False
txtHysteresis.Visible = False
End Sub
```

```
Private Sub Timer1_Timer()
intVerandering = (intVerandering * 30) / 60
```

```
On Error Resume Next
```

```
If txtKr < 0 Then
    txtKr = 0
End If
If txtKr > 100 Then
    txtKr = 100
End If
dblKr = Val(txtKr)
```

```
If txtTi < 0 Then
    txtTi = 0
End If
If txtTi > 100 Then
    txtTi = 100
End If
dblTi = Val(txtTi)
```

```
If txtTd < 0 Then
    txtTd = 0
End If
If txtTd > 100 Then
    txtTd = 100
End If
dblTd = Val(txtTd)
```

```
If txtHysteresis < 1 Then
```



```

    txtHysteresis = 1
End If
If txtHysteresis > 7 Then
    txtHysteresis = 7
End If

lblPv.Caption = CInt(dblProceswaarde) & "°C"
dblError = intSetpoint - dblProceswaarde
lblError.Caption = CInt(dblError)

If optPid.Value = True Then
    If dblProceswaarde < 50 Then
        dblProceswaarde = dblProceswaarde + intToevoer
    End If
    If dblProceswaarde > -10 Then
        dblProceswaarde = dblProceswaarde - intVerandering
    End If
    pid
End If

If optOnOff.Value = True Then
    aan_uit
End If

pctGrafiek.Cls
intPvgrafiek(100) = dblProceswaarde
For intX = 0 To 99
    intPvgrafiek(intX) = intPvgrafiek(intX + 1)
    pctGrafiek.PSet (intX, 70 - (intPvgrafiek(intX)))
Next intX

pctGrafiek.Line (0, 70 - intSetpoint)-(100, 70 - intSetpoint), vbYellow
End Sub



---


Private Sub hscSetpoint_Change()
If hscSetpoint.Value > 0 Then
    lblSetpoint2.ForeColor = &HFF&
Else
    lblSetpoint2.ForeColor = &HC00000
End If
intSetpoint = hscSetpoint.Value
lblSetpoint2.Caption = hscSetpoint.Value & "°C"
End Sub



---


Private Sub pid()
intFilter = 10
dblInputD = dblProceswaarde + (dblVorige - dblProceswaarde) * (dblTd / 60)
dblVorige = dblProceswaarde

dblInputDF = dblInputDF + (dblInputD - dblInputDF) * intFilter / 60

```

```

dblOutput = (intSetpoint - dblInputDF) * (dblKr / 100) + dblFeedback
If dblOutput > 100 Then
    dblOutput = 100
End If
If dblOutput < 0 Then
    dblOutput = 0
End If
intVerandering = 100 - dblOutput
dblFeedback = dblFeedback - (dblFeedback - dblOutput) * dblTi / 60

If optDuister.Value = True Then

    If dblProceswaarde > intSetpoint Then
        imgDoorsnede.Picture = LoadPicture(App.Path & "\verwarming_min_licht_aan.bmp")
    End If

    If dblProceswaarde < intSetpoint Then
        imgDoorsnede.Picture = LoadPicture(App.Path & "\verwarming_veel_licht_aan.bmp")
    End If

    If dblProceswaarde = intSetpoint Then
        imgDoorsnede.Picture = LoadPicture(App.Path & "\verwarming_aan_licht_aan.bmp")
    End If

End If
If optLicht.Value = True Then

    If dblProceswaarde > intSetpoint Then
        imgDoorsnede.Picture = LoadPicture(App.Path & "\verwarming_min_licht_uit.bmp")
    End If

    If dblProceswaarde < intSetpoint Then
        imgDoorsnede.Picture = LoadPicture(App.Path & "\verwarming_veel_licht_uit.bmp")
    End If

    If dblProceswaarde = intSetpoint Then
        imgDoorsnede.Picture = LoadPicture(App.Path & "\verwarming_aan_licht_uit.bmp")
    End If

End If

End Sub

Private Sub aan_uit()
intSetpoint = hscSetpoint.Value
intHysteresis = Val(txtHysteresis)
intOnder = intSetpoint - intHysteresis
intBoven = intHysteresis + intSetpoint
If optLicht.Value = True Then

```

```

If (dblProceswaarde < intOnder) Or (dblProceswaarde > intBoven) Then
    If dblProceswaarde >= intBoven Then
        dblProceswaarde = dblProceswaarde - 1
        imgDoorsnede.Picture = LoadPicture(App.Path & "\verwarming_uit_licht_aan.bmp")
    End If

    If dblProceswaarde <= intOnder Then
        dblProceswaarde = dblProceswaarde + 1
        imgDoorsnede.Picture = LoadPicture(App.Path & "\verwarming_aan_licht_aan.bmp")
    End If
Else
    If dblProceswaarde = (intOnder) Then
        intA = 1
    End If

    If dblProceswaarde = (intBoven) Then
        intA = 2
    End If

    Select Case intA
        Case 1
            dblProceswaarde = dblProceswaarde + 1
            imgDoorsnede.Picture = LoadPicture(App.Path & "\verwarming_aan_licht_aan.bmp")
        Case 2
            dblProceswaarde = dblProceswaarde - 1
            imgDoorsnede.Picture = LoadPicture(App.Path & "\verwarming_uit_licht_aan.bmp")
    End Select
End If
End If

If optDuister.Value = True Then

    If (dblProceswaarde < intOnder) Or (dblProceswaarde > intBoven) Then
        If dblProceswaarde >= intBoven Then
            dblProceswaarde = dblProceswaarde - 1
            imgDoorsnede.Picture = LoadPicture(App.Path & "\verwarming_uit_licht_aan.bmp")
        End If

        If dblProceswaarde <= intOnder Then
            dblProceswaarde = dblProceswaarde + 1
            imgDoorsnede.Picture = LoadPicture(App.Path & "\verwarming_aan_licht_aan.bmp")
        End If
    Else

        If dblProceswaarde = (intOnder) Then
            intA = 1
        End If

        If dblProceswaarde = (intBoven) Then
            intA = 2

```

```

End If

Select Case intA
    Case 1
        dblProceswaarde = dblProceswaarde + 1
        imgDoorsnede.Picture = LoadPicture(App.Path & "\verwarming_aan_licht_aan.bmp")
    Case 2
        dblProceswaarde = dblProceswaarde - 1
        imgDoorsnede.Picture = LoadPicture(App.Path & "\verwarming_uit_licht_aan.bmp")
End Select
End If
End If
End Sub

```

mdlVariabelen :

```

Option Explicit
Global intWaarde As Integer
Global intToevoer As Integer
Global intVerandering As Integer
Global intSetpoint As Integer
Global intTaal As Integer
Global intPvgrafiek(100) As Integer
Global intFilter As Integer
Global intX As Integer
Global intHysteresis As Integer
Global intOnder As Integer
Global intBoven As Integer
Global intA As Integer

Global dblProceswaarde As Double
Global dblKr As Double
Global dblTi As Double
Global dblTd As Double
Global dblError As Double
Global dblFeedback As Double
Global dblVorige As Double
Global dblOutput As Double
Global dblInputD As Double
Global dblInputDF As Double
Global dblInput As Double

Global strBeaufort As String

```