

Assignment 1

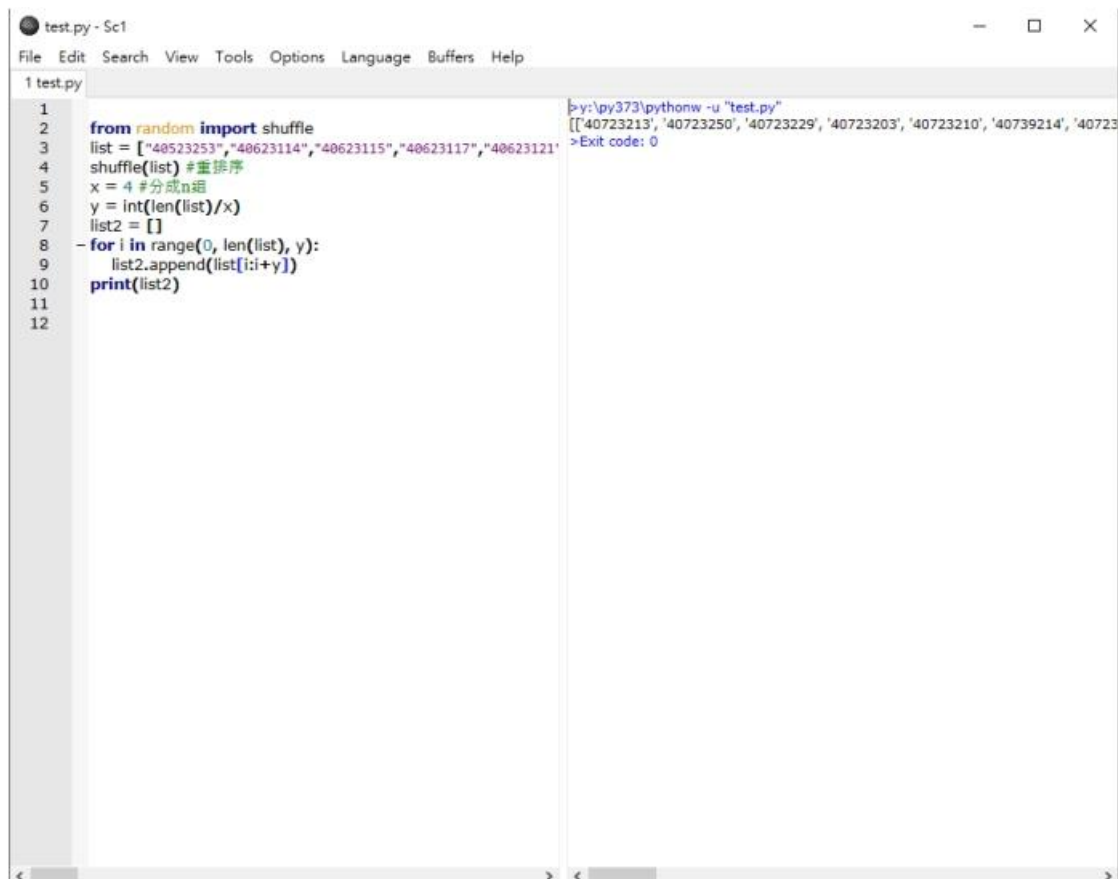
1.請描述如何針對該課程進行有效的隨機分組，或者隨機進行點名？

利用 python 的模組

1. `random`(隨機函數):可以從指定的項目中抽取隨機的元素。
2. `shuffle`(隨機排序):可以將 `list` 中的元素隨機排序，不能單獨導入，必須先從 `random` 導入。

隨機分組：

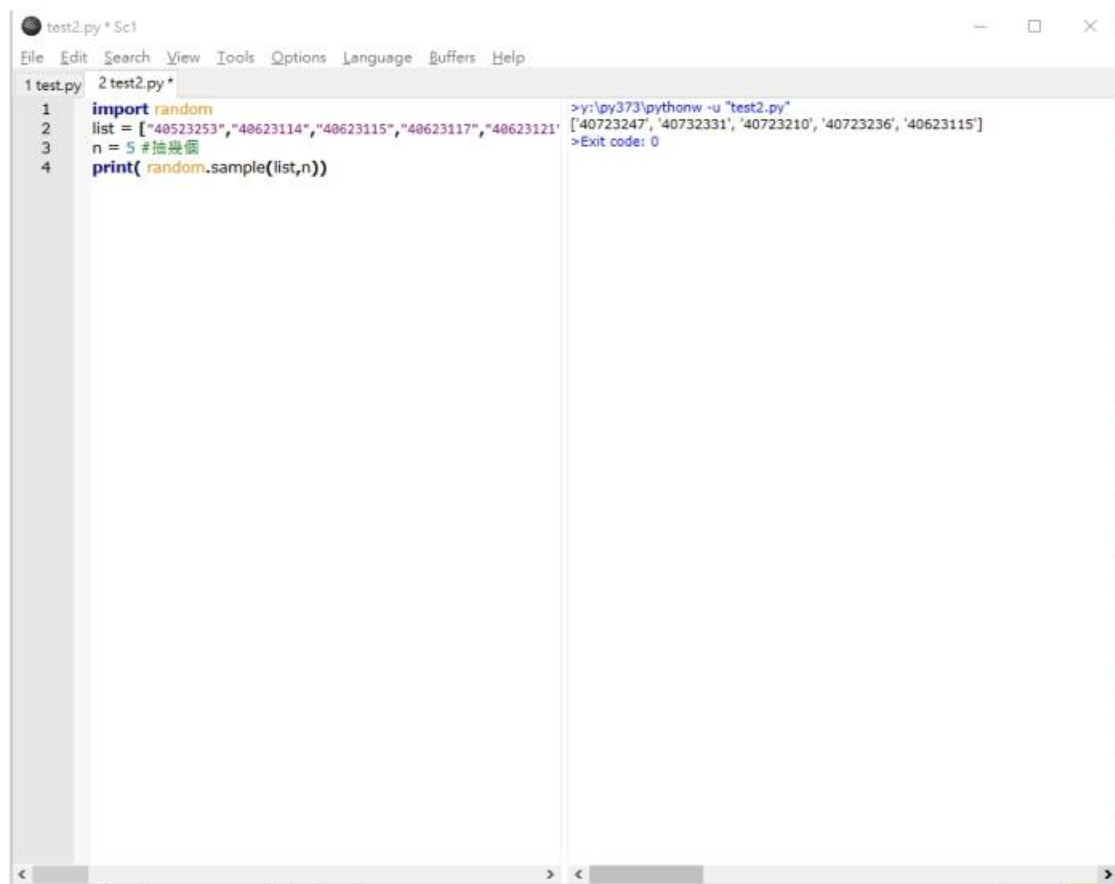
先從學校取得選課人員的資料，由 `python` 需要將名單的各個學號當作字串才可以用，我找不到在程式上直接做轉換的方法，就利用記事本的取代功能，將空格取代成","，在丟進程式裡，再利用 `shuffle` 的隨機排序，指定 `x` 為總組數，利用 `len` 知道名單有多少人，再除以組數，就知道每組應有多少人，並利用 `int` 指定為整數，用 `for` 迴圈並且限制每組可容納的數量，就能分組了。



```
test.py - Sc1
File Edit Search View Tools Options Language Buffers Help
1 test.py
1
2 from random import shuffle
3 list = ["40523253","40623114","40623115","40623117","40623121"
4 shuffle(list) #重排序
5 x = 4 #分成n組
6 y = int(len(list)/x)
7 list2 = []
8 for i in range(0, len(list), y):
9     list2.append(list[i:i+y])
10 print(list2)
11
12
b> y:\py373\pythonw -u "test.py"
[["40723213", "40723250", "40723229", "40723203", "40723210", "40739214", "40723
> Exit code: 0
```

隨機進行點名：

導入 random 模組，指定抽取人數為 n，利用 random.sample 隨機抽取 n 個。



```
test2.py * Sc1
File Edit Search View Tools Options Language Buffers Help
1 test.py 2 test2.py *
1 import random
2 list = ["40523253", "40623114", "40623115", "40623117", "40623121"]
3 n = 5 #抽幾個
4 print(random.sample(list,n))

>y:\py373\pythonw -u "test2.py"
['40723247', '40732331', '40723210', '40723236', '40623115']
>Exit code: 0
```

資料來源：

<https://codertw.com/>

<https://www.runoob.com/>

心得：

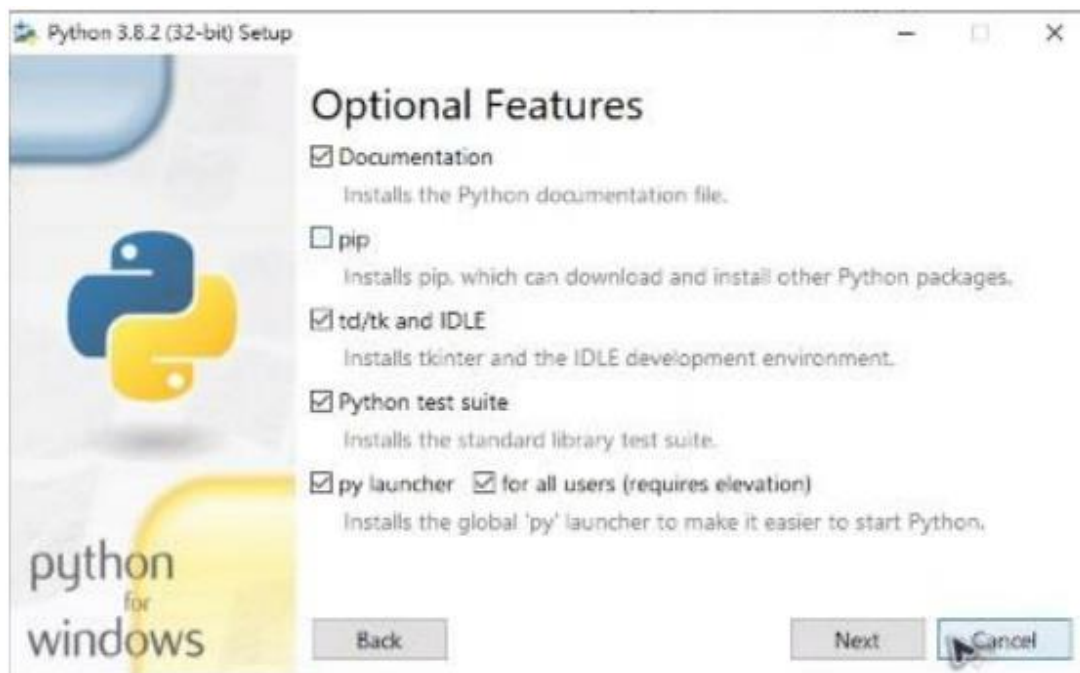
要製作隨機抽點系統需要去了解許多程式語言與模組，第一次去了解程式語言，感覺有點困難，但是只要了解那個邏輯，其實沒有到非常困難。

2.重新創建 python 3.8.2可攜系統

在新的資料夾內安裝 python3.8.2 在/data 下，資料夾命名為 py382，並陸續安裝 MSYS2、PortableGit、SciTE。

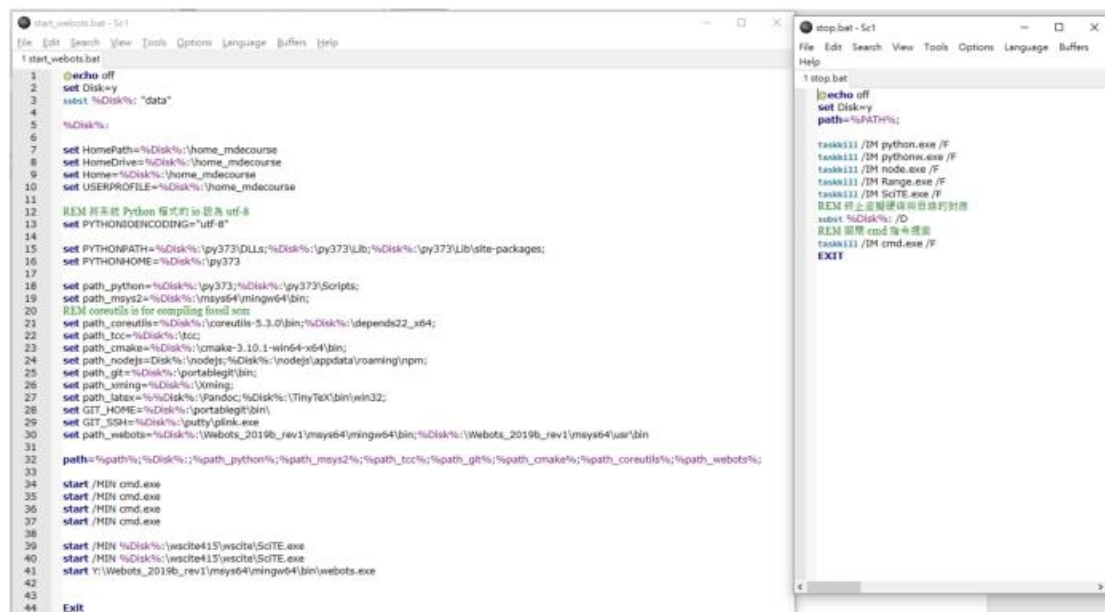


Python3.8.2 安裝時，pip 的選項必須取消，之後再到網站上載下檔案，方進 data 下，之後再安裝。



現在先製作開啟的批次檔，之後打開 start.bat，利用黑窗打上指令安裝

python 缺少的 mod。



```
start_webots.bat
1 start_webots.bat
2
3 @echo off
4 set Disk=y
5 subst %Disk%: "%data"
6
7 %Disk%:
8
9 set HomePath=%Disk%\home_mdecourse
10 set HomeDrive=%Disk%\home_mdecourse
11 set Home=%Disk%\home_mdecourse
12 set USERPROFILE=%Disk%\home_mdecourse
13
14 REM 將系統 Python 編碼的位元組改為 utf-8
15 set PYTHONENCODING=utf-8
16
17 set PYTHONPATH=%Disk%\py373\DLLs;%Disk%\py373\Lib;%Disk%\py373\Lib\site-packages;
18 set PYTHONHOME=%Disk%\py373
19
20 set path_python=%Disk%\py373;%Disk%\py373\Scripts;
21 set path_msys2=%Disk%\msys64\mingw64\bin;
22 REM coreutils is for compiling fossil scm
23 set path_coreutils=%Disk%\coreutils-3.3.0\bin;%Disk%\depends22_x64;
24 set path_tcc=%Disk%\tcc;
25 set path_cmake=%Disk%\cmake-3.10.1-win64-x64\bin;
26 set path_nodejs=%Disk%\nodejs;%Disk%\nodejs\appdata\roaming\npm;
27 set path_git=%Disk%\portablegit\bin;
28 set path_xming=%Disk%\Xming;
29 set path_latex=%Disk%\Pandoc;%Disk%\TinyTeX\bin\win32;
30 set GET_HOME=%Disk%\portablegit\bin;
31 set GET_SSH=%Disk%\putty\plink.exe
32 set path_webots=%Disk%\Webots_2019b_rev1\msys64\mingw64\bin;%Disk%\Webots_2019b_rev1\msys64\user\bin
33
34 path=%path%;%Disk%;;%path_python%;%path_msys2%;%path_tcc%;%path_git%;%path_cmake%;%path_coreutils%;%path_webots%;
35
36 start /MIN cmd.exe
37 start /MIN cmd.exe
38 start /MIN cmd.exe
39 start /MIN cmd.exe
40
41 start /MIN %Disk%\wsrte415\wsrte\ScTE.exe
42 start /MIN %Disk%\wsrte415\wsrte\ScTE.exe
43 start Y:\Webots_2019b_rev1\msys64\mingw64\bin\webots.exe
44
45 Exit
```

```
stop.bat - Sc1
1 stop.bat
2
3 @echo off
4 set Disk=y
5 path=%path%;
6
7 taskkill /IM python.exe /F
8 taskkill /IM pythonw.exe /F
9 taskkill /IM node.exe /F
10 taskkill /IM Range.exe /F
11 taskkill /IM ScTE.exe /F
12 REM 將上述路徑與目錄對應
13 subst %Disk%: /D
14 REM 關閉 cmd 指令提示
15 taskkill /IM cmd.exe /F
16
17 Exit
```

3. What do you need to know from

<http://www.coppeliarobotics.com/helpFiles/index.htm> to

implement a fourwheeled robot?

Simulation&Simulation settings dialog:

Simulation 在說明模擬程式的功能，主要可以分成即時模擬與非即時模擬，並且說明兩種模擬運行的概要，在即時模擬下根據個人電腦的運算效能有限，並不是可以在各種情況下使用；在即時模擬的狀況下，如果覺得時間過久，也可以隨時調整模擬速度。

Simulation settings dialog 則是在解講模擬速度中可調節的選項。

BubbleRob tutorial:

這個章節則是在介紹如何建立機器人，設定各物件與感測器，在上學期的課程上，已經有實際操作過的經驗，對於此章節相對較熟悉。

Line following BubbleRob tutorial:

這章節主要是在擴展 BubbleRob 的功能讓它遵循地上的線，利用感測器對著地面感應，輸入程式碼使他能夠跟著線走。

External controller tutorial:

除了上述的方法，這裡還提供額外 7 種方式去控制機器人，並且提供了教程給使用者學習。

2.4.3 物件排列示範（圖 2.3）

到目前為止，在我們關於定位兩個對象（一堵牆和一個對象）的討論中，僅從兩個方面簡化了討論。我們將擴展為三個本節中的尺寸。讓我們以對象排列為例進一步。讓我們看一下在機櫃中定位兩個對象的幾種方法看看我們有什麼選擇。出於本示例的目的，假設兩個物體都是“磚”（字面上是磚），其大致尺寸為：

- 2.5 英寸厚
- 3.5 英寸寬
- 8.0 英寸長

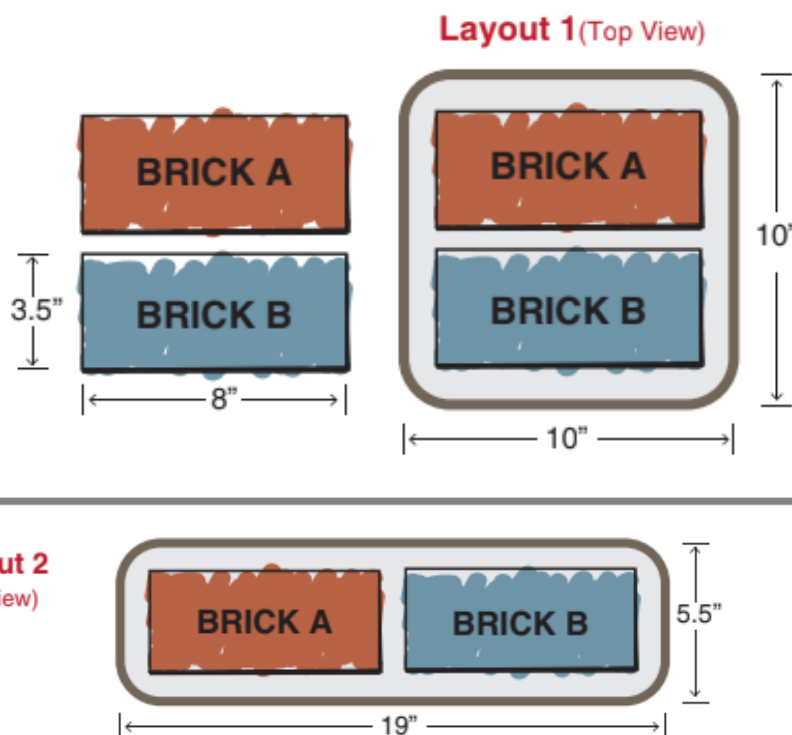
在我們的 2D 示例中，我們將忽略“厚度”，而僅使用 3.5x8.0 寬度和長度尺寸。因此，我們基本上有一個 3.5x8.0 的矩形。（我們將進一步回到 3D 示例，因為這為我們添加了更多選擇。）請參閱圖 2.3。

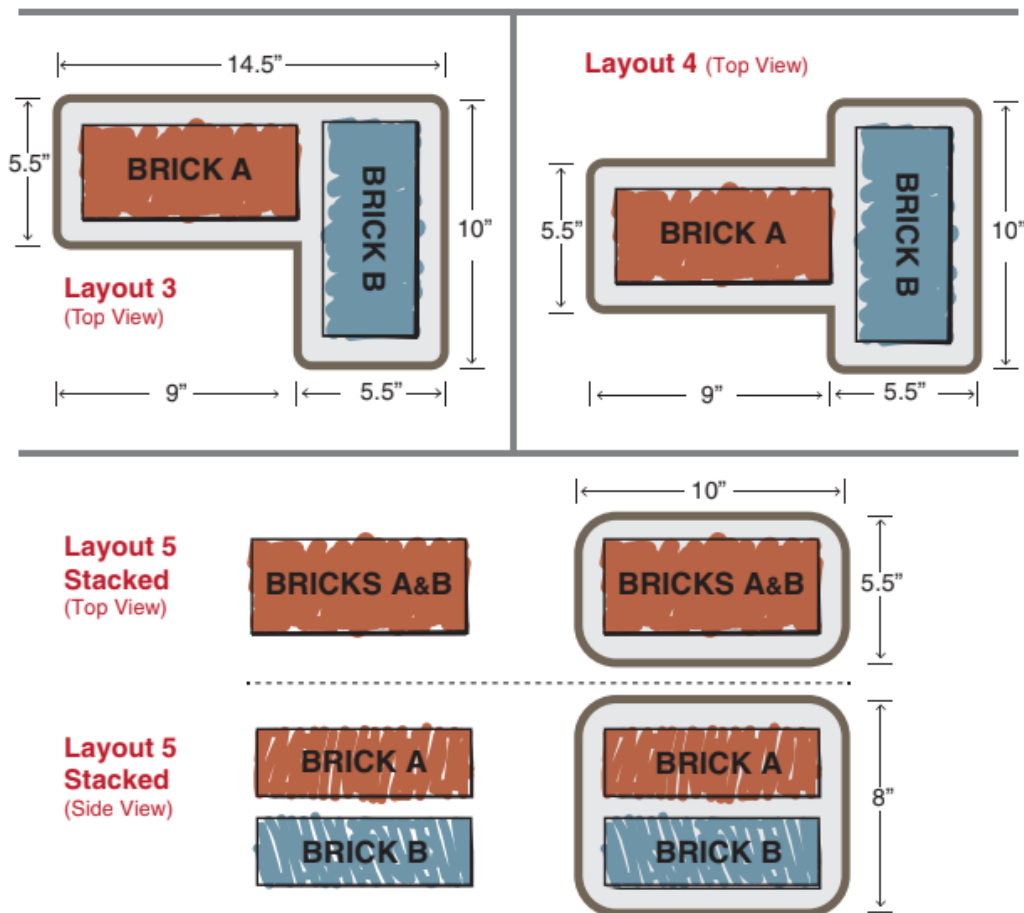
現在，假設我們設計的基本出發點是容納兩塊磚，外殼中的 A 磚和 B 磚（尺寸相同）。在這個初始一點，我們沒有以下限制：

- 外殼的整體尺寸或形狀
- 外殼材料成本

最佳物件放置

在外殼中的兩個對象（磚 A 和磚 B）的不同配置的例子





我們可以輕鬆地設想（至少）找到磚 A 和磚 B 相對於彼此，產生非常不同的外殼。當然有超過五種不同的方式，但我選擇了標準的“笛卡爾”排列相互平行或對齊的磚塊。讓我們看看這五個不同的佈局並評論一下為什麼可能會有一些優勢（在其他佈局上）。磚 A 和磚 B 之間假定恆定為 1 英寸，並且假定磚和牆（側面、上、下）。

- 版式 1：沿著寬度並排放置磚 A 和磚 B
- 佈局 2：磚 A 和磚 B 沿長度對齊
- 佈局 3：“L 形”的磚 A 和磚 B
- 版式 4：“T 形”的磚 A 和磚 B
- 版式 5：“堆疊”磚 A 和磚 B（3D 版本 - 這是唯一的版式使用了“三維”

現在，讓我們分析五種佈局。（對於所有佈局，我們假設外殼的薄皮，使外殼的寬度，長度，和高度。此外，我們將忽略機櫃可能具有的圓角，並假設為方形角）。版式 1 似乎是最簡單的版式，以相對方形結束用於外殼。所得的機箱高度為 $10 \times 10 \times 4.5$ 。六個方面（區域）是 $2 \times (10 \times 10) + 4 \times (10 \times 3.5)$ 。

佈局 2 是“長”，而不是“正方形”。這種外殼可能具有獨特的應用程序，例如更好地利用辦公桌空間。產生的附件是 $19 \times 5.5 \times 4.5$ 高。六個邊（面積）是

$2 \times (19 \times 5.5) + 2 \times (19 \times 3.5) + 2 \times (5.5 \times 3.5)$ 。

- 佈局 3 可能更適合“轉角”應用程序。產生的附件是 $(5.5 \times 10 \times 4.5) + (5.5 \times 9 \times 4.5)$ 。八個邊（面積）為 $3.5 \times (4.5 + 9 + 10 + 4.5 + 5.5 + 14.5) + (9 \times 5.5 \times 2) + (10 \times 5.5 \times 2)$ 。

- 版式 4 與版式 3 相似，但外觀更加對稱（相同的體積並且與佈局 3 的周長相同）。

- 佈局 5 提供最小的“平面圖”，但導致最高的佈局。最終的外殼高 $10 \times 5.5 \times 8$ 。六個邊（面積）為 $2 \times (10 \times 5.5) + 2 \times (10 \times 8) + 2 \times (5.5 \times 8)$ 。

這些簡單的佈局說明，即使在此放置兩個對象，案例，兩塊磚，代表了很多可能性。如果要添加第三個對象或不同大小的物體，人們可以看到這變得相當複雜。有時，有一個根本原因是一個對象相對於另一個對象的放置，因為一個對象的“輸入”應該靠近另一個對象的“輸出”（對象的嵌套）。在無論如何，讓我們繼續在上面討論這五個佈局的一些相對優點。在佈局之間進行選擇可能會有基本的美學原理。

也就是說，可以選擇佈局 1，因為它被認為是“更直接”（更多“誠實”），那麼佈局 3 可能會被認為更“有趣”。所以，選擇佈局的選擇可能取決於客戶將找到一個特殊的外殼形狀更令人愉悅（因此導致產品）。

如何優化（最小化）外殼的表面積 - 哪種佈局導致最小的表面積？同樣，外殼包圍兩塊磚的側面，頂部和底部。

	Volume (Cubic inch)	Perimeter area (Total square inches)
Layout 1	450	340
Layout 2	470.25	380.5
Layout 3	470.25	377
Layout 4	470.25	377
Layout 5	440	358

關於物件排列的一些結論

如果要最小化重量，則佈局 1 最好，因為重量主要是由於機櫃外圍區域。（磚塊在所有佈局中的重量都相同，空氣是可以忽略不計。）

如果要最小化體積，則佈局 5 是最好的。這可能對一些尺寸受限制的設計。

佈局 2、3 和 4 的音量相同。

最大的外圍區域是佈局 2。迭代“佈局 6”（未顯示）會是一個封閉的球體。半徑為 6 英寸的球體可以容納佈局 5 產生的體積約為 900 立方英寸，而只有一個周長約 450 平方英寸的面積 - 也許是“創意”解決方案？

顯然，隨著添加更多“對象”，優化該對象會變得更加困難。

體積和周長的對象。設計師的獨創性帶有“嵌套”在給定的佈局區域中“繪製”各種幾何對象。可以使用各種技術來優化佈局的緊湊性。全部一次確定外殼中的對象後，設計人員可以對這些對象進行建模並開始將它們放置在以下方向上：

1. 有效利用空間。
2. 將需要彼此靠近的物體放置在盡可能近的位置其他。這可能是由於機械，熱或電氣原因造成的。例如，如果電纜連接兩個對象，則具有這兩個對象越近越好—如何通過連接來消除電纜將對象 1（直接）連接到對象 2？
3. 放置需要彼此保持盡可能遠的距離的物體，彼此流血。同樣，可能出於機械，熱或電氣方面的原因。

應該注意的是，佈局 3 和佈局 4 可能會更複雜，生產（製造）。非對稱外殼可能更難比直牆要好。如果外殼是工具，則不一定如此 2 建立設計（模製或鑄造）產品。但是，如果外殼是金屬板，則多餘的壁提供了更多的製造問題。

關於 3D 設計的注意事項：由於我們設計的所有對象實際上都是 3D 對象，我們實際上將需要進行 3D 設計。快速以 2D 草圖可以解決設計意圖的一部分，但所有細節都需要在設計中顯示移至 3D。這使得在 CAD 中工作對於精度和速度至關重要在當代設計界是必需的。創建所有對象的 3D 模型使檢查間隙和配合變得容易進行更改和優化。一個例子這就是機械設計師如何看待印刷電路板組件（PCBA）的方式。它基本上具有 2D 佈局，其中許多組件安裝在印刷電路上板。但是，所有這些組件都處於不同的高度，因此間隙 PCBA 之上和之下的變化在很多方面都不同。因此，PCBA 之上和之下的變化在很多方面都不同。因此，PCBA 雖然最初被認為是“2D 區域”（處於“草圖階段”），但具有“厚度”，即使其成為 3D 體積。

章節總結

本章從設計的起點出發，我們只有一個理念。它向我們展示瞭如何將這個想法轉化為對象的幾何位置進入太空，使我們對該想法進行了物理體現。

我們首先查看起點並定義邊界設計—我們從什麼開始，什麼是設計的“外緣”。我們有定義客戶需要的產品。

我們看到了設計是如何從版本 1 進行到版本 X 的，其中 X 是提供我們認為是客戶所需的設計。

最後，我們看了看如何將各個對象放入可以優化設計以解決客戶需求。需要權衡考慮，我們必須意識到我們如何確定最佳選擇這些權衡。

4.6 成型塑料

電子外殼的設計師必須對塑料模具具有紮實的知識，過程以及如何設計塑料零件。大多數工程學位都不會這項技能非常受重視，因此在工作中很可能會達到並磨練。關於塑料零件的正確設計的文獻很多，我將提供一些很好的參考資料，其中很多來自塑料供應商（原始顆粒）本身。塑料供應商自己已經學到了很多多年來使用樹脂的交易，每個人都可以從中受益經驗。另外，參考文獻格倫·比爾（Glenn L. Beall）的著作[6]可以被視為“聖經”。Beall 是該領域的知名專家。我將提供“前十名”的列表塑料設計準則，但這只是眾多清單的重點每個設計師都可以添加的內容。

塑料零件設計人員可能的#1能力是了解工具-將會用於他們自己的部分並了解哪些選項是提供注塑工具。了解注塑模具後，模具，以下六個概念將有助於注塑成型的設計零件（有關更多信息，請參見參考文獻[7]）：

（見圖 4.6）

1. 拔模的想法需要將零件從模具中彈出。
 2. 將塑料“注入”到模具中的主澆口（或“澆口”）的位置。這個門的位置（以及隨後的“下降”需求）將是一個很大的考慮因素-零件所需化妝品的配方。門控可以在邊緣或模具的型芯（反向澆口）或型腔側。
 3. “模具流動”的概念需要很好地理解。需要倒圓角角，通常是等厚度的設計，並且對肋骨高度的限制是突出顯示。隨著熔體的冷卻，它從粘性液體變成半固體，最終，成為堅實的一部分。填充零件的區域更加困難離零件澆口最遠。
 4. 再次顯示脫衣舞酒吧的位置，突出了化妝品表面的需要。
 5. 如果零件需要“底切”，則模具將顯示如何進行加工。達到的效果（以及使模具複雜化的程度）。
 6. 模具中所示的模具“分型線”反映了零件設計從而顯示出這些設計的困難。
- 以上工具設計功能是從經驗中學到的。

在所有情況下，該部分設計需要由工具設計人員審核（通常由設計人員進行審核或簽約）模具製造商），零件設計師應獲得保證，他們的設計是“模具-能”以一種簡單的方式。零件設計和工裝都需要認真對待由設計人員和項目管理團隊進行審查，因為模具具有：

1. 大量的資本支出（K \$）
2. 交貨時間很長（需要數週才能完成）
3. 困難的（時間和金錢）修訂過程

模製塑料零件的最大優點之一是它不需要的第二個零件操作成為成品零件（與“類似”金屬零件相比）。通常，模製塑料部件不需要在外部進行美容噴漆。大多數（cus-塑料零件最終在外部具有模壓紋理通常通過用“紋理圖案”

蝕刻模具來實現。注意這個模具紋理會導致零件的底切（通過添加少量即可實現部分的草稿）。

成型零件需要一些常用的第二種操作（這些操作是在成型操作之外完成的操作，通常是由成型者完成的（或由成型商外包））：

1. 在模具本身認為不可行的零件上需要的孔或切口 - 有時更容易通過第二步操作而不是通過工具添加孔作為模具的一部分。因此，可以增加鑽孔或攻絲操作。
2. 如果門位於裝飾區域，則可能需要機加工以修復“門標記”。如果澆口區域不是裝飾性表面，也可以用手移除澆口（並且該表面不需要機加工）。
3. 插入件（金屬，螺紋緊固件）可通過超聲波方式放入零件（或這些嵌件也可以模製在內。）
4. EMI 屏蔽層可應用於塑料部件。這採取各種形式：
 - 繪畫
 - 電鍍
 - 金屬屏蔽層的粘接
5. 粘接操作（超聲波或膠粘劑）可熔合一個或多個模製零件一起。
6. 可以進行各種裝飾（絲網印刷，繪畫）。模內可以進行操作以這種方式合併圖形。
7. 必須明確確定上述任何操作所需的所有夾具，進行成本核算，並製定合理的時間表。

以上所有第二項操作都需要在結束時清楚地標出零件文檔，其中應包括任何質量控制可接受標準。這些第二次操作會大大增加零件的單價因此，它們需要成為整個設計過程的一部分，設計者明確提出的設計。第二個操作需要取消與成型商深入探討，發現它們在大多數情況下都是可以實現的划算的方式。

零件的塑料選擇：設計人員必須檢查先前列出的 15 種特性（對於任何材料），以決定選擇哪種樹脂。蠕變數據（長期粘彈性行為）對塑料也很重要。其他一些獨特的方面包括：

1. 成型工藝有很多：注射成型/吹塑成型/壓縮成型/熱成型（壓力或真空成型）/反應注射成型（RIM）。我主要講的是注塑工藝。
2. 卡扣配合要求在應力與應變方面具有特定限制的材料。
3. 存在與塑料顆粒供應商共同選擇塑料的數據（參見參考文獻[8]）。這些供應商還可以審查設計並可能指示如果您的應用程序需要定制的解決方案。供應商很大有關材料和成型的信息庫。正如之前的“承諾”一樣，以下是我對適當塑料的十大建議零件設計。

可以使用常駐的模具填充“程序”來分析大多數設計在內部或在模塑商處。這些程序採用設計的 CAD 幾何數據，建議的材料，並確定優化過程以正確填充模具：

1. 必須盡一切努力保持壁厚均勻。較厚的區域可能會導致被稱為“水槽”的外觀缺陷。大排骨可能會導致壁厚的部分，肋的底部與主壁相遇（這可以限制肋骨的“高度”）。這也是螺絲釘的常見現象老闆在基地與主牆相遇的地方。螺絲凸台也會招致額外的複雜性，因為凸台中的插入物需要所需的壁厚結構要求。似乎牆（大於標稱）的地方需要厚度-希望通過在其中添加功能來“挖空”大區域該工具將導致（更）均勻的壁厚。永遠想到分為三個維度。通常，設計是一系列 2D 截面，用於達成目的。但是，塑料零件的輪廓可以會導致彎曲區域，從而導致壁厚不均勻的區域。
2. 出於所有原因，半徑所有過渡區域。這些使熔體流動更容易圓角部分。而且，只能通過模具加工來產生尖角這本身就很尖銳。這些鋒利的邊緣很難維持工具的壽命。半徑內的區域應與半徑外的區域“匹配”，即內半徑+標稱壁厚=外半徑（零件的半徑）。的內徑與壁厚之比（ R / T ）應盡可能高減小橫截面應力，建議 $R / T > 0.75$ 。一些注意事項在塑料零件圖上包括“半徑所有角 0.030 半徑或指定範圍”；但是，必須注意任何預期的注意事項覆蓋圖形上的所有事件。
3. 選擇工具製造商/模具對零件的整體滿意度至關重要。其他因素（“成本”除外）也進入了此選擇，例如：
 - 工具質量 - 工具的材料選擇和熱處理。
 - 工具的交付時間表 - 首件和生產數量零件。
 - 溝通路徑的開放性 - 供應商與溝通的程度零件設計師，有關零件設計的建議交流，以及任何可能出現的持續性問題。定期安排的里程碑發布切片是不可避免的。面對面的討論可以幫助交流，但是互聯網也可以得到有效利用。
 - 零件的總成本 - 包括交付（模製/盒裝/裝運），任何夾具零件，輔助操作和質量控制所需的。
 - 從當前批准的供應商中選擇供應商 - “合格”的成本必須考慮新的供應商。另外，目前批准的問題供應商已經眾所周知，而未經批准的供應商可能意味著未知的問題。
4. 設法減少零件所需的二次加工量。看在涉及工具滑軌的權衡中。零件中的特徵通過向工具中添加幻燈片（或工具中的旋轉刀片）產生的減少第二次操作的方法。但是，這些滑片或旋轉刀片使工具更昂貴並使成型操作複雜化。考試-例如，如果零件中需要大螺紋，則可以在工具內形成這些螺紋通過包括工具的旋轉部分（“鬆開”以移除零件）從工具中刪除），從而消除了（輔助）線程操作（在外部完成工具）。但是，將螺紋作為第二螺紋可以較便宜而不是將線程包含在更複雜的工具（“活動”）。零件中的某些孔可以通過以下方式在工具中形成被稱為“交叉關閉”。這種交叉關閉會“刺穿”帶有碎片的側壁匹配一個角度。但是，交叉封鎖的負面影響是“階梯式”工具中的分型線會使工具稍微複雜化。再次，討論與工具供應商/模具一起決定折衷選擇的價值是無價的。
5. 吃水應盡可能大方，而不影響均勻的牆面厚度或設計。在某些地區，零吃水

是可能的－請與鑄工！（例如，紋理化實際上會產生不需要的“底切”如果吃水深度允許零件“彈開”，則側面拉動。）

所需的草稿具有以下功能：

- 材料
- 收縮
- 零件設計需求
- 粗糙度

肋骨的最小牽伸允許肋骨更高，同時保持相對肋與壁相遇處的壁厚均勻。

6. 放寬設計中的加強筋數量。這是其中之一使用模製塑料零件的優勢在於可以加強零件的肋骨增加的開銷很少。這些肋骨增加了剛度和控制力熔體流動。應探索肋的圖案，例如圓形和矩形。相互連接的肋或側壁可增加零件的整體剛度。

7. 注意零件上“焊接線”的確切位置。焊接線發生在材料在零件內流動並在對面（自身）相遇的地方塑料最初進入零件（在工具澆口處）的側面。這些焊縫線導致零件的結構薄弱區域（因為材料在兩個熔合路徑的接合點）。而且，焊接線顯示為實際的“線”在零件的外部產生了必須解決的美學問題。

8. 考慮零件設計中的區域，這將導致產生非常大的區域工具本身的薄部分（甚至是彎曲的芯銷）。這些薄工具部分可能難以維護，並且破損可能會延遲零件成型。始終在模具上添加“部分”，以便於移除。這些部分（工具刀片）可用於以下任一方面：

- 易於維護
- 輕鬆產生另一個零件變化，例如，帶有孔的零件和沒有孔的部分

例如，將零件修訂級別放在工具上是很常見的插入，以便隨著修訂級別的更改，僅（小）插入需要被修改。

9. 考慮將特徵添加到設計中，這將有助於對齊（或在結構上幫助）的配合部分。例如，在可以配合的零件中添加圓形凹槽帶有圓形正極的零件可以幫助配合的整體設計這兩個部分的配對。

10. 考慮脫模針的位置以及“多少”。的脫模針將在零件已經（大約）後幫助從模具中彈出零件固化。但是，他們通常會在（希望）非部分的裝飾面。這些位置應由設計師審查確保它們放置在最佳位置。例如，脫模針應請勿將其放置在設計墊片的位置（因為墊片表面會不平坦）。

11. （是，在前十名中排名第11位。）同時搜索文獻和其他文獻塑料零件以獲取知識。多年來，許多“部落知識”已在以下領域獲得：

- 按扣設計
- 塑料零件的緊固過程（組裝方法）和緊固件
- 新材料混合
- 模內裝飾

通用的塑料設計過程遵循以下六個步驟：正確的修訂版此處假定文檔的內容－請參見第1章。關於文檔12。設計審查的廣泛含義是包括應該參與審查的所有人員各個部分和整個項目的周期（應包括內部審閱者，承包商和供應商）：

1. 零件設計。設計經過審查。
2. 放置工具投標。已發送CAD文件（視情況而定）。設計經過審查由工具供應商和樹脂製造商提供。對設計的任何更改都會再次出現已審查。同意開發工具的時間表和第一篇文章。計劃同意接受第一篇文章。已知零件的所有價格（包括同意進行夾具安裝）。
3. 最終設計獲得批准；工具採購訂單已下達特定訂單（並且同意）修訂級別。
4. 工具製造商輸入CAD文件並添加正確的收縮係數（零件將收縮以確保正確的零件幾何形狀。
5. 定期安排會議以審查項目里程碑。更新給項目管理。零件和/或時間表的任何更改都明確廣播。
6. 撰寫並審閱第一篇文章。考慮的修改包括對項目進度的影響。零件的預生產運行計劃為適當。

4.7 鑄造金屬

鑄造金屬與上一章關於塑料成型的部分有很多共同之處。但是，有一些明顯的區別。首先，如上節所述實際上有很多塑料成型工藝，可以說是一樣的用於鑄造過程。有很多鑄造技術。鑄造技術通常會有所不同：

- 金屬鑄件
- 正常鑄造零件的尺寸範圍
- 公差應按工序確定
- 工具成本
- 單價
- 表面光潔度期望
- 建議的最低吃水深度
- 正常最小截面厚度
- 訂購數量
- 正常的交貨時間

常見的鑄造技術如下所述（參考文獻[10]，另外參考[11，12]）：

- 壓鑄：將熔融金屬在壓力下注入硬化的鋼模中，經常用水冷卻。模具打開，鑄件彈出。
- 永久模具：將熔融金屬重力倒入鑄鐵模具中，並進行塗層用陶瓷模具清洗。芯可以是金屬，沙子，沙殼或其他。模具打開並彈出鑄件。新的LPPM方法壓力可高達15 psi。
- 投資（失蠟）：金屬模具製造蠟或塑料複製品。這些是澆注，然後用包紮材料包起來，烘烤，然後倒入金屬在合成腔中。模具破裂以去除鑄件。

- 石膏模具：將石膏漿倒入半模上並使其凝固；然後從模型中取出模具，烘烤並組裝，然後將金屬倒入合成腔。模具破裂以去除鑄件。
- 陶瓷模具：將陶瓷漿料倒在上模和下模上，允許固化，然後將模具從模型上移開，並烘烤（在 1800°F 下），從而產生堅硬，穩定的模具。模具組裝有或沒有芯和金屬倒入合成腔。模具破裂以去除鑄件。
- 石墨模具：類似於陶瓷模具，但使用石墨模具代替。芯銷通常是鋼。
- 樹脂外殼模具：將樹脂塗層的砂子倒在熱金屬圖案上，進行固化切成殼狀的半模。這些已從花樣中移除，用或組裝沒有核心。將金屬倒入所得腔中。模具破損去除鑄件。
- 砂型鑄造：將調質好的沙子包裝到木材或金屬半模上，從圖案上取下，組裝有芯或無芯，然後將金屬倒入產生的空腔。可以使用各種芯材。黴菌破碎成鑄件。現在正在使用的專用粘合劑可以改善公差和表面完。

Table 4.2 Table of casting techniques (from [6])

Technique	Metals	Size range	Tolerances (inch)	Surface finish (rms)
Die casting	Al/zinc/Mg	<2 ft ²	0.001–2	32–63
Permanent mold	Al/zinc/brass	Oz – 100 lbs.	0.015 basic	150–250
Investment	Most castable	Oz – 150 lbs.	0.003 basic	63–125
Plaster mold	Al/zinc/brass	<3 ft ²	0.005 basic	63–125
Ceramic mold	Most castable	<350 lbs.	0.005 basic	80–125
Graphite mold	Zinc or zinc-Al	Oz – 10 lbs.	0.005 basic	63–125
Resin shell mold	Most castable	<4 ft ²	0.008–10 basic	125–350
Sand casting	Most castable	Oz and up	0.03 basic	150–700
Metal injection mold	Ferrous	Under ¼ lbs.	0.003 basic	45
Powder metal	Ferrous/SS/Al	20 in ²	0.004 basic	16–90

- 金屬注射成型：極細的金屬粉末與粘合劑材料結合被注入模具。零件彈出，粘合劑熔化或溶解，並抽真空燒結，理論密度為 94–99%。
- 金屬粉末：金屬粉末在移動上部之間的模具桶中壓縮下拳。下沖頭彈出零件，然後進行燒結和定尺寸需要嚴格的公差。

下表 4.2 總結了上述各種鑄件的區別技術。

鑄造零件的設計通常具有與注射相似的約束模製零件，即在所有角處都需要恆定的壁厚和半徑，但是有一些重要的區別。由於大多數鑄造材料較脆，它們更容易在尖角處產生應力，因此需要更大的半徑。注射成型產生的下沉不是鑄件的大問題。怎麼樣-以往，零件仍需要去芯以保持恆定的壁厚（並減輕重量）。

鑄造工藝雖然類似於注塑成型，但可帶來獨特的效果必須解決的問題：

A. 氣孔鑄件不是均勻的。鑄件的“表皮”（外部層（可能為 0.020 英寸厚）相對光滑且連續。結果因為與“熔融”流相比，模具表面本身相對較冷金屬，並且在該材料的外邊緣上形成了一層緻密的表皮。在下面該表皮可能是被困在金屬中相對較深的空氣（氣體）區域“多孔的。”如果用機器在其外表面上進行

切割，則孔隙率區域（金屬不均勻）會被暴露。有時會在機器外部進行機器切割鑄件，因為任何特定尺寸的公差都可能很高。對於例如，可以鑄造標稱尺寸為 4.000 英寸的外部尺寸在 ± 0.010 英寸處。因此，如果鑄件最終達到 4.010 英寸，則需要 4.000 英寸在設計中，可能需要 0.010 才能將鑄件進行機械加工。機加工 0.010 折部件的一部分可能會帶走堅硬的皮膚，從而露出潛在的孔隙。

而且，如果“鑄態”表面對設計而言不夠“光滑”，設計需要低的表面粗糙度（對於用於壓力的表面密封）。如果“鑄態”表面粗糙度為 500μ 英寸且表面粗糙度為 32μ 英寸（對於例如）需要提供一個密封表面（帶有 O 形圈），問題將需要機械加工。有時會使用浸漬劑來提高壓力密封性和光滑的表面壓鑄件中的面孔。

使用厭氧劑和甲基丙烯酸酯的系統是指定浸漬時很少使用。這些系統生產密封鑄件準備進行壓力測試。鑄件中的孔隙率也會造成嚴重的質量控制問題。

由於孔隙可以是鑄造過程的結果（正確設置溫度和壓力，加上原料的稠度），甚至在某個點上也會在一部分中產生孔隙儘管批次中的某些零件沒有孔隙。如果零件中出現孔隙，則在負載下的臨界面，零件可能會失效。因此，必須過濾孔隙率為此，必須對過程進行控制和監視。如果該零件是“任務-關鍵”部分涉及安全或國防需要，則該部分將需要一些用於確保零件強度的方法。

B. 所需的二次操作鑄件通常需要一些“大型”操作鑄造過程後進行準備，以便最終組裝。這些運作可以分為機加工和裝飾/防腐蝕塗料。與注塑零件類似，材料“進入”零件的區域（按“劑量”）必須除去。也就是說，需要進行加工操作以從注料口上清除殘留在零件上的材料的痕跡。

這種機加工操作可能像手動推剪一樣簡單，可以去除殘留物但是，如果此痕跡位於美觀的表面上，則可能會更加詳細。不像注塑零件，鑄造過程通常還涉及“修剪模具”去除模具分型線上的材料，這些材料要么以“飛邊”的形式脫落，要么是故意的，允許盟友出去用固體金屬填充整個零件。這個“修剪死”可以由壓機和治具組成，以控制修整過程，或者說是“取消控制”旋轉（例如“手動”完成的砂輪）。“閃光”是由於微小上模和下模之間不匹配。上下模具在哪裡“聚在一起”，鑄造過程的壓力迫使少量（0.001 - 0.005 英寸）的材料要“噴出”，並且隨著模具的使用越來越多，這種情況變得更糟。修整模具（和夾具）的成本可能高達數千美元。避免用手修剪。因此，這是要根據實際情況做出的另一項成本/時間決定處理任何工具支出。

可以將多種表面處理系統應用於壓鑄件，以提供裝飾效果，防腐蝕或增加硬度和耐磨性。建議向供應商尋求有關表面處理的具體信息和參考文獻，因為它們隨鑄造材料和表面處理目的而變化。例如對於鑄鋁：

- 可以通過油漆，拋光/環氧樹脂，電鍍和粉末來實現裝飾效果塗層。
- 可以通過油漆，陽極氧化，鉻酸鹽，和虹彩。

- 可以通過浸漬來填充和密封表面/亞表面。
- 硬質陽極氧化可提高耐磨性。

沒有鑄件設計者應該沒有 Ref 的參考材料。 [13]。NADCA 壓鑄產品規範標準是“參考書”所需的背景信息包括：

- 工藝和材料選擇
- 壓鑄模具
- 合金數據
- 工程和設計（特定於壓鑄件）
- 質量保證
- 商業慣例

心得：

在設計工件時，一開始種會隨意地畫除草圖，再去想如何製造，這個文章提供給我們在設計時，應該是去思考要注意哪些方面，並且提供很有力的示範，讓我們明白，為什麼要這樣。

11

實例探究

閱讀完本章後，讀者將可以：

1. 實驗他在設計和實時方面的知識機電系統的實施。
2. 進行從 A 到 Z 的機電系統設計。
3. 能夠執行機電一體化設計的不同階段系統。
4. 能夠解決控制問題並建立控制律我們必須實時實施。
5. 能夠使用 C 語言的中斷概念編寫程序實時實施。

11.1 引言

在前面的章節中，我們開發了一些概念，說明了它們的應用。通過學術實例向讀者展示這些結果如何應用。更多具體來說，我們已經了解瞭如何設計機

電一體化系統，介紹了成功完成所需設計所必須遵循的不同步驟機電系統。我們已經介紹了我們必須在設計：

- 機械部分
- 電子線路
- C 語言中用於實時執行的程序

這些工具已用於一些實際系統，並提供了更多詳細信息幫助讀者執行自己的設計。

對於控制算法，我們介紹的大多數示例都是學術性的具有完美的模型。不幸的是，對於實際的系統，我們將擁有只是可以在某些特定條件下描述系統的實現，並且由於某些原因，該模型無法在實際操作中按預期運行，算法的時間執行。這可能是由於不同的忽視動態可能會改變某些頻率的行為。

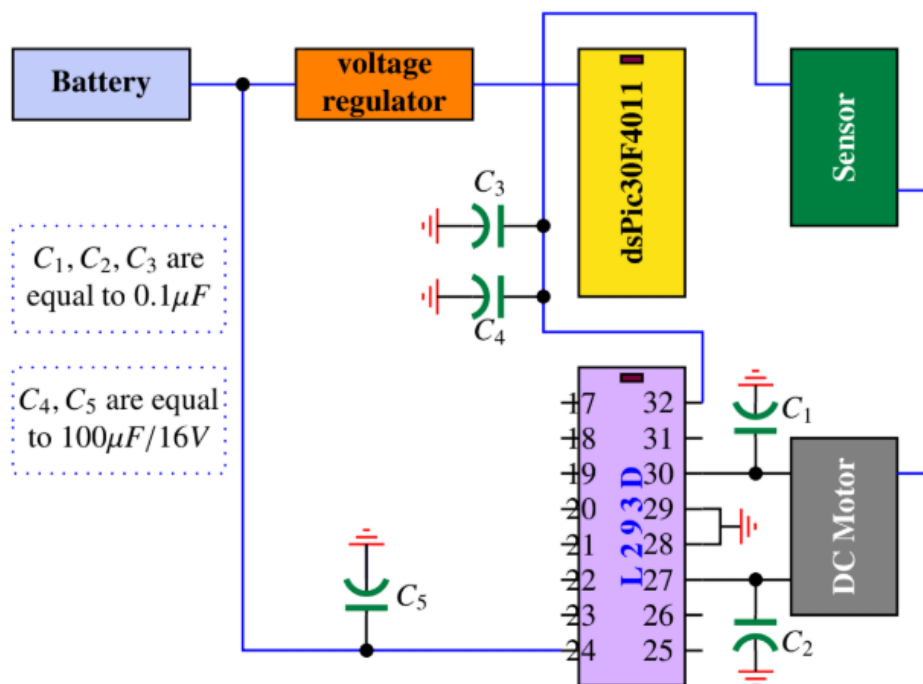
本章的目的是向讀者展示我們如何實時實施我們在前幾章中為實際應用開發的理論結果系統。我們將逐步進行並顯示所有步驟，以簡化操作過程。讀者。我們在本章中考慮的案例研究是討論和在前幾章中進行了設計。

11.2 直流電動機套件的速度控制

作為第一個示例，讓我們考慮驅動直流電動機的直流電動機的速度控制，機械部分。該示例的選擇非常重要，因為大多數系統將使用這種直流電動機。我們將考慮的直流電動機由 Maxon 製造公司。該電動機非常重要，因為它帶有齒輪箱（比率 6：1），並且編碼器，每轉給出一百個脈衝，每轉給出 600 個脈衝通過使用正交方法將其發展到 2000 年的革命每轉四百個脈衝。在此示例中，我們使用的系統如果更靈活，我們將在前面介紹控制算法的實時實現並提供更多優勢。

該電動機的數據表給出了所有重要參數，因此容易獲得該執行器的傳遞函數。我們正在考慮的負載在這個例子中是一個帶有刻度的小磁盤，我們想控制速度然後就位。這種設置在圖 1 和 2 中示出。 11.1- (11.2)。磁盤我們他們認為直徑等於 0.06 m，質量等於 0.050 Kg。用這些數據和直流電動機的數據表之一，我們可以獲得轉移磁盤速度和輸入電壓之間的函數。

首先讓我們專注於負載的速度控制。在這種情況下建立該系統的傳遞函數（直流電動機執行器及其負載）可以使用數據表，磁盤上的信息以及 Boukas [1]中的結果或繼續進行識別。使用第一種方法得出第二章的結果在[1]中，我們得到：



$$G(s) = \frac{K}{\tau s + 1}$$

$$\begin{aligned} K &= 48.91 \\ \tau &= 63.921 \text{ ms} \end{aligned}$$

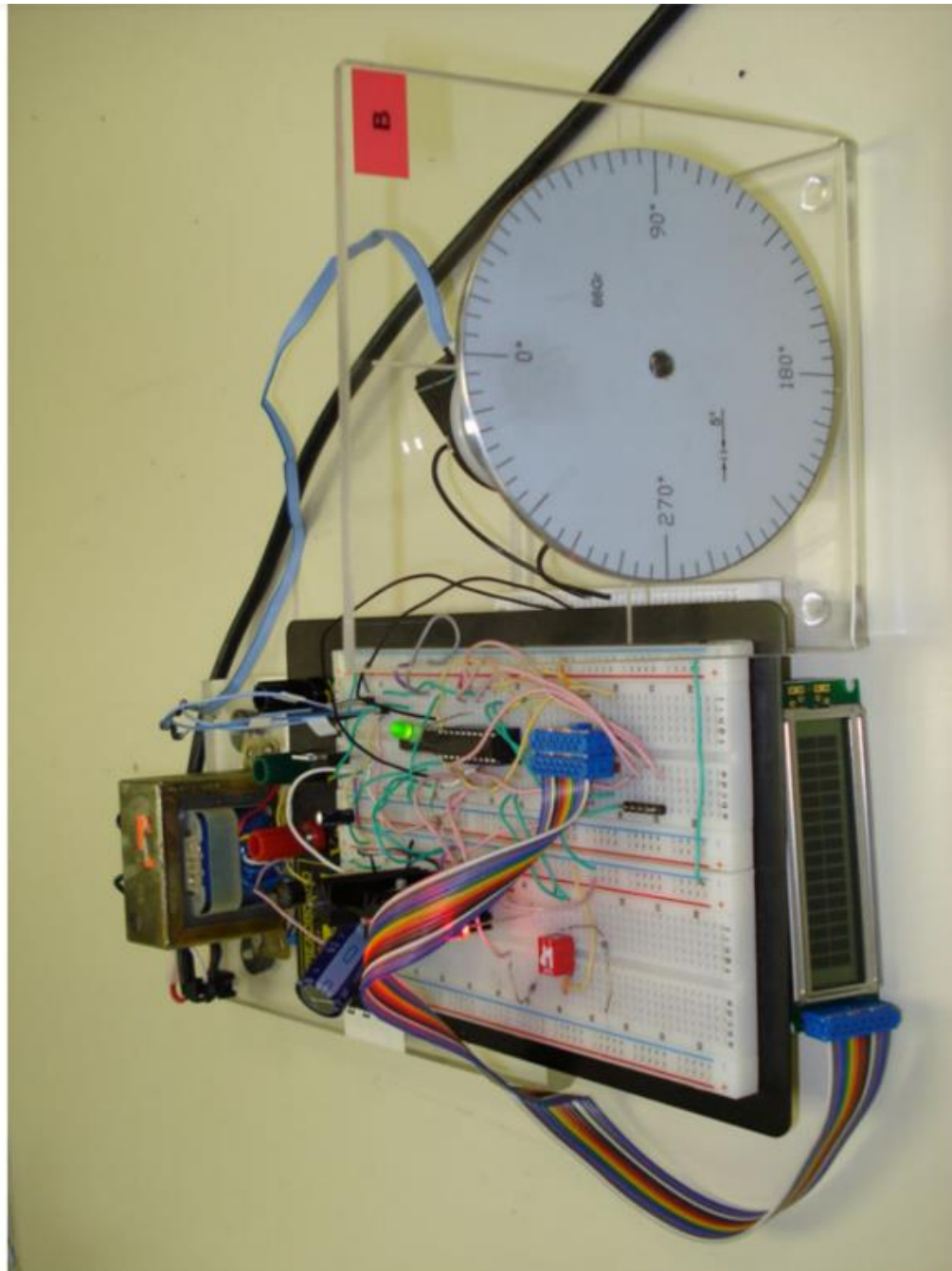


Fig. 11.2 Real-time implementation setup

根據系統傳遞函數的表達式以及所需的性能要處理該結果，我們至少需要一個比例和積分器（PI）控制器。

該控制器的傳遞函數由下式給出：

$$C(s) = K_P + \frac{K_I}{s}$$

其中要確定 K_P 和 K_I 的增益以迫使負載具有我們強加的表演。

使用零序持有者和 Z-transform 表，我們得到：

$$\begin{aligned} G(z) &= \frac{Kz(1 - e^{-\frac{T}{\tau}})(1 - z^{-1})}{(z - 1)(z - e^{-\frac{T}{\tau}})} \\ &= \frac{K(1 - e^{-\frac{T}{\tau}})}{z - e^{-\frac{T}{\tau}}} \end{aligned}$$

對於控制器，使用梯形離散化，我們得到：

$$\begin{aligned} C(z) &= \frac{U(z)}{E(z)} = K_P + K_I \frac{T}{2} \frac{z + 1}{z - 1} \\ &= \frac{(K_P + \frac{TK_I}{2})z + (-K_P + \frac{TK_I}{2})}{z - 1} \end{aligned}$$

將分子和分母除以 z 並回到時間，我們得到：

$$u(k) = u(k - 1) + \left(K_P + \frac{TK_I}{2}\right)e(k) + \left(-K_P + \frac{TK_I}{2}\right)e(k - 1)$$

結合執行器的傳遞函數及其負載和其中之一控制器，我們得到以下閉環傳遞函數：

$$F(z) = \frac{K(1 - e^{-\frac{T}{\tau}})(K_P + \frac{TK_I}{2})z + K(1 - e^{-\frac{T}{\tau}})(-K_P + \frac{TK_I}{2})}{z^2 + (K(K_P + \frac{TK_I}{2})(1 - e^{-\frac{T}{\tau}}) - 1 - e^{-\frac{T}{\tau}})z + K(1 - e^{-\frac{T}{\tau}})(-K_P + \frac{TK_I}{2}) + e^{-\frac{T}{\tau}}}$$

現在使用所需的表演，很容易得出結論，極點是

$$s_{1,2} = -\zeta\omega_n \pm j\omega_n \sqrt{1 - \zeta^2}$$

其中 ζ 和 ω_n 分別代表阻尼比和固有頻率我們系統控制的閉環。

根據控制理論（參見 Boukas [1]），眾所周知，超調 $d\%$ 和 5% 的建立時間 t_s 由下式得出：

$$\begin{aligned} d\% &= 100e^{\frac{-\pi\zeta}{\sqrt{1-\zeta^2}}} \\ t_s &= \frac{3}{\zeta\omega_n} \end{aligned}$$

$$\zeta = 0.707$$

$$\omega_n = \frac{5}{\tau\zeta} = 110.6387 \text{ rad/s}$$

$$s_{1,2} = -78.2216 \pm 78.2452j$$

$$z_{1,2} = 0.5317 \pm 0.2910j$$

$$\begin{aligned}\Delta_d &= (z - 0.5317 - 0.2910j)(z - 0.5317 + 0.2910j) \\ &= z^2 - 1.0634z + 0.3674\end{aligned}$$

$$1 + C(z)G(z) = \Delta_d$$

$$K_I = \frac{0.3040}{KT \left(1 - e^{-\frac{T}{\tau}}\right)}$$

$$K_P = \frac{-0.4308 + 2e^{-\frac{T}{\tau}}}{2K \left(1 - e^{-\frac{T}{\tau}}\right)}$$

備註 11.2.1 在這種情況下必須謹慎，因為我們不在乎傳遞函數零的位置，因此我們可能會有一些實現此控制器時會感到驚訝。顯然，我們的表演將獲得（建立時間和超調量）取決於零的位置。有關此問題的更多詳細信息，請向讀者介紹 Boukas [1]。

現在要實施此 PI 控制算法並確保所需的性能我們將使用 Microship¹ 的微控制器。這個選擇是由於我們的經驗使用這種類型的微控制器。讀者可以記住，任何其他微型其他製造商的控制器稍作改動即可完成工作。在這個例如，我們將使用 Microhip 的 dsPIC30F4011 單片機。我們實現的代碼使用 C 語言編寫。

這種語言是為簡單起見而採用。該實現具有以下結構：

```
//
// Put here the include
//

#include "p30F4011.h"          // proc specific header

//
```

¹ See www.microchip.com

```

// Define a struct
//
typedef struct {
    // PI Gains
    float K_P;      // Propotional gain
    float K_I;      // Integral gain
    //
    // PI Constants
    //
    float Const1_pid; //  $K_P + T K_I/2$ 
    float Const2_pid; //  $-K_P + T K_I/2$ 
    float Reference;  // speed reference

    //
    // System variables
    //
    float y_k;  // y_m[k] -> measured output at time k
    float u_k;  // u[k]   -> output at time k
    float e_k;  // e[k]   -> error at time k

    //
    // System past variables
    //
    float u_prec; // u[k-1] -> output at time k-1
    float e_prec; // e[k-1] -> error at time k-1

}PIStruct;

PIStruct thePI;

thePI.Const1= thePI.K_P+T*thePI.K_I/2;
thePI.Const2=-thePI.K_P+T*thePI.K_I/2;
thePI.Reference=600;

//
// Functions
//
float ReadSpeed(void);

float ComputeControl(void);

float SendControl(void);

//
// Interrupt program here using Timer 1 (overflow of counter Timer 1)
//
void __ISR _T1Interrupt(void)    // interrupt routine code
{

```

```

// Interrupt Service Routine code goes here
float Position_error;

//
// Read speed
//
thePI.y_m=ReadSpeed();

thePI.e_k= thePI.Reference-thePI.y_m;

//
// Compute the control
//
ComputeContrl();

//
// Send control
//
SendControl();

IFS0bits.T1IF=0;    // Disable the interrupt
}

int main ( void )      // start of main application code
{
// Application code goes here
int i;

// Initialize the variables Reference and ThePID.y_m (it can be read
// from inputs) Reference = 0x8000; // Hexadecimal number
// (0b... Binary number) ThePID = 0x8000;

// Initialize the registers
TRISC=0x9fff; // RC13 and RC14 (pins 15 and 16) are configured as outputs
IEC0bits.T1IE=1; // Enable the interrupt on Timer 1

// Indefinite loop
while (1)
{
}
return 0
}

% ReadSpeed function
int ReadSpeed (void)
{
}

```



```

% ComputeControl function
int ComputeControl (void)
{
thePI.u_k=thePI.u_prec+thePI.Const1*thePI.e_k+thePI.Const2*thePI.e_prec;
}

% SendControl function
int Send Control (void)
{
sendControl()

//
// Update past data
//
thePI.u_prec=thePI.u_k;
ThePI.e_prec=thePI.e_k;
}

```

從該結構可以看出，首先我們注意到系統將進入循環，並在每次中斷時調用函數：

- ReadSpeed；
- ComputeControl；
- SendControl；

並採取適當的措施。

ReadSpeed 函數將在每個採樣時間返回加載速度，由 ComputeControl 函數使用。

SendControl 函數發送通過 L293D 芯片向執行器提供適當的電壓。使用編譯器 HighTec C 獲取十六進制代碼，並使用 PicKit-2 上傳該文件在微控制器的存儲器中。

有關如何獲取十六進制的更多詳細信息代碼，我們邀請讀者閱讀編譯器 HighTec C 或編譯器的手冊 Microchip 的 C30。在這種情況下，國家方法是微不足道的，我們將不發展它。

11.3 直流電機套件的位置控制

讓我們專注於負載位置控制。遵循與負載矢量類似的步驟上一節中開發的位置控制，我們首先需要選擇所需的我們希望系統具有的性能。以下表演是施加：

- 系統穩定在閉環狀態；
- 穩定時間 t_s 為 2%，等於我們可以擁有的最佳時間
- 超調等於 5%
- 步進功能作為輸入的穩態等於零

使用性能和傳遞函數，很容易得出結論：

比例控制器 KP 足以滿足這些性能。

在此示例中，我們將使用連續時間方法來設計

控制器。在上一章的基礎上，我們的系統模型如下：

$$G(s) = \frac{K}{s(\tau s + 1)}$$

where K and τ take the same values as for the speed control.

Let the transfer controller be give by:

$$C(s) = K_P$$

Using these expression, the closed-loop transfer function is given by:

$$\begin{aligned} F(s) &= \frac{C(s)G(s)}{1 + C(s)G(s)} \\ &= \frac{\frac{KK_P}{\tau}}{s^2 + \frac{1}{\tau}s + \frac{KK_P}{\tau}} \end{aligned}$$

Since the system is of type 1, it results that the error to a step function as input is equal to zero with a proportional controller.

Based on the specifications, the following complex poles:

$$s_{1,2} = -\zeta w_n \pm j w_n \sqrt{1 - \zeta^2}$$

will do the job and the corresponding characteristic equation is given by:

$$s^2 + 2\zeta w_n s + w_n^2 = 0.$$

Equating this with the one of the closed-loop system we get:

$$\begin{aligned} 2\zeta w_n &= \frac{1}{\tau} \\ w_n^2 &= \frac{KK_P}{\tau}. \end{aligned}$$

To determine the best settling time t_s at 2 %, notice that we have:

$$t_s = \frac{4}{\zeta w_n}.$$

Using now the fact:

$$\zeta w_n = \frac{1}{2\tau}$$

因此，此控制器在%2 處的最佳建立時間為 8 倍系統的恆定時間。小於可獲得的任何值。其實這是如果我們在改變 KP 時關注閉環系統的根源，那麼這是微不足道的。這個由圖 11.3 給出。為了固定控制器的增益，所需極點 $s_{1,2} = -7.5 \pm j$ ，我們使用該圖並選擇 $\zeta = 0.707$ 。得出 $K_P = 0.1471$ 。

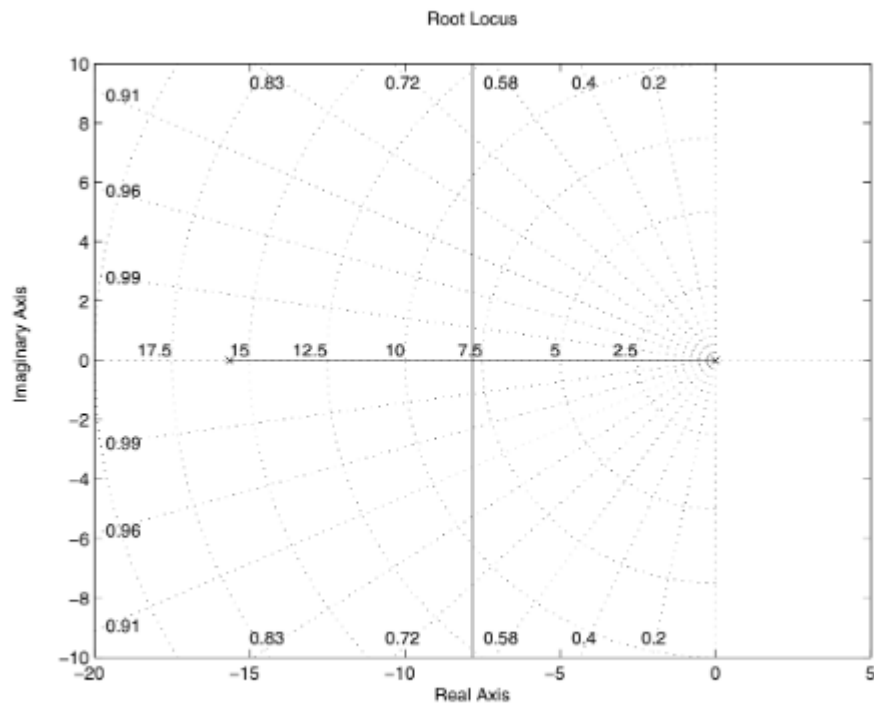


Fig. 11.3 Root locus of the dc motor with a proportional controller

使用該控制器，幅度為的階躍函數的時間響應等於 30 度由圖 11.4 表示，由此我們可以得出結論：控制器以%2 的穩定時間滿足所有期望的性能至 0.5115 s。但是，如果我們實現此控制器，實際情況將與由於變速箱的齒隙不包括在使用的模型中因此，實時結果將有所不同，並且誤差永遠不會為零。至為了克服這個問題，我們可以使用比例和微分控制器在%2 處提供更好的建立時間。給出該控制器的傳遞函數通過：

$$C(s) = K_P + K_D s$$

其中 K_P 和 K_D 是要確定的增益。

備註 11.3.1 重要的是要注意，使用比例和衍生控制器將在閉環傳遞中引入零，這可能會改善放置時間是否合適。根據其位置，超調和穩定時間將受到影響。有關此問題的更多詳細信息，請參閱讀者至[1]。

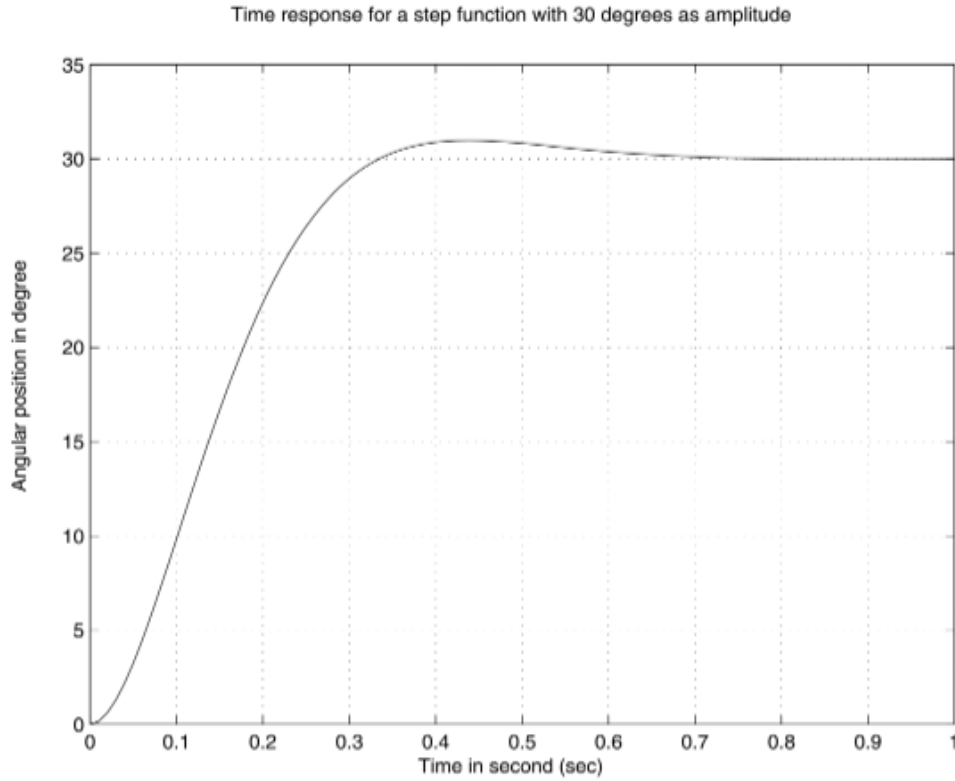


Fig. 11.4 Time response for a step function with 30 degrees as amplitude

With this controller the closed-loop transfer function is given by:

$$\begin{aligned}
 G(s) &= \frac{C(s)G(s)}{1 + C(s)G(s)} \\
 &= \frac{\frac{KK_D s + KK_P}{\tau}}{s^2 + \frac{1 + KK_D}{\tau}s + \frac{KK_P}{\tau}}
 \end{aligned}$$

和以前一樣，控制器的設計使用了兩個複雜的極點。要是我們將得到的兩個特徵方程式等價：

$$\begin{aligned}
 2\zeta w_n &= \frac{1 + KK_D}{\tau} \\
 w_n^2 &= \frac{KK_P}{\tau}
 \end{aligned}$$

在這種情況下，我們有兩個未知變量 K_P 和 K_D 以及兩個代數唯一確定增益的方程。它們的表達式如下：

$$K_P = \frac{\tau w_n^2}{K}$$

$$K_D = \frac{2\tau\zeta w_n}{K}$$

現在使用期望的性能，我們得出與之前類似的結論。輸入的穩態誤差等於幅度等於 30 度的階躍函數實例等於零，並且阻尼比 ζ 對應於過衝等於 %5 等於 0.707。穩定時間， t_s 為 %2，我們可以將其固定為系統時間常數的比例，得出：

$$w_n = \frac{4}{\zeta t_s}.$$

現在，如果將穩定時間固定為 3τ ，我們將得到：

$$w_n = 29.4985.$$

使用這些值，我們可以得到以下控制器增益值：

$$K_P = 1.1374$$

$$K_D = 0.0545.$$

它給出了以下複雜的兩極：

零為：

$$z = -20.8618.$$

使用該控制器，輸入幅度等於的時間響應在圖 11.5 中表示了 30 度。從該圖可以看出超調和建立時間少於使用比例獲得的控制器。實施比例或比例和微分控制我們需要獲得控制律的遞推方程。以此目的，我們需要使用不同的方法離散化控制器的傳遞函數 ods 較早提出。讓我們使用梯形方法替換 s 乘 2

$C(z) = K_P$ for the proportional controller

$C(z) = K_P + K_D \frac{2}{T} \frac{z-1}{z+1}$ for the proportional and derivative controller

$z + 1$ 用於比例和微分控制器如果我們通過控件分別表示 $u(k)$ 和 $e(k)$ 以及引用之間的誤差以及瞬時 kT 的輸出，我們得到以下表達式：

1. 比例

$$u(k) = K_P e(k)$$

Time response for a step function with 30 degrees as amplitude

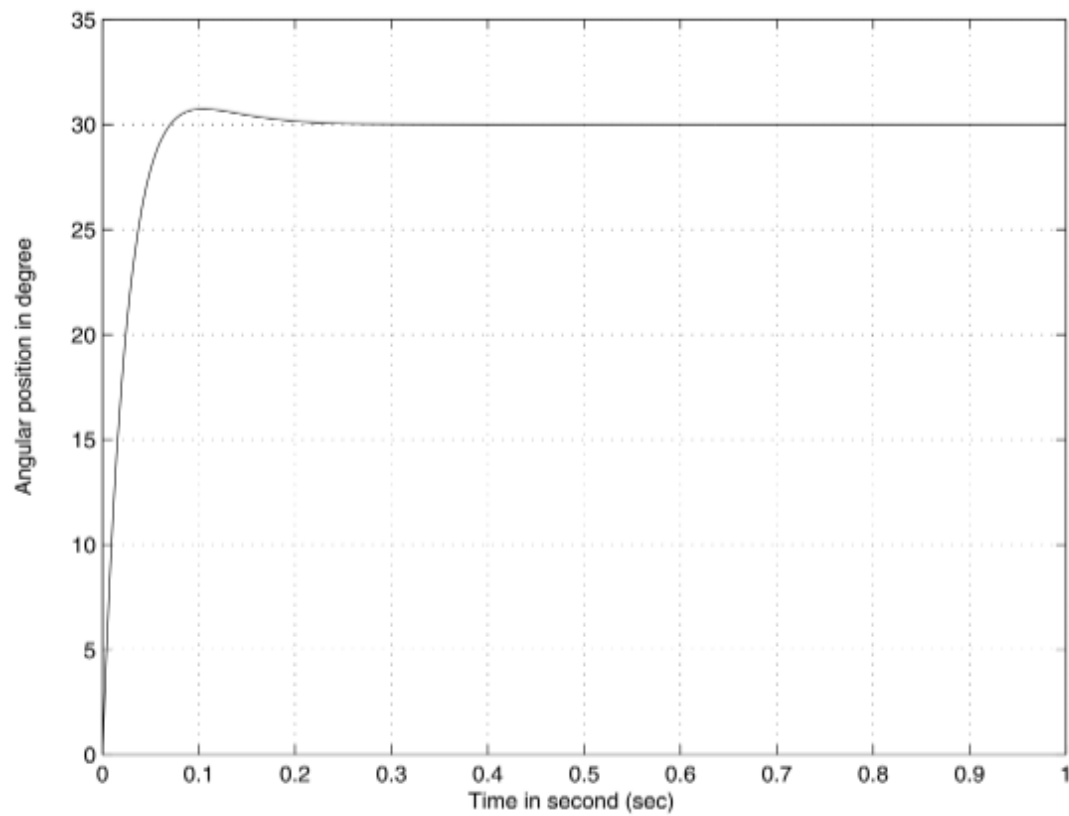


Fig. 11.5 Time response for a step function with 30 degrees as amplitude

2. for the proportional and derivative controller

$$u(k) = -u(k-1) + K_P + \frac{2K_D}{T}e(k) + K_P - \frac{2K_D}{T}e(k-1)$$

The implementation is this controller uses the same function with some minors changes. The listing the corresponding functions is:

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Main program                                     %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

main

% Data

% Variables

% While loop while (1) do
    ReadSpeed;
```

```

        ComputeControl;

        SendControl; end;

% ReadSpeed function

% ComputeControl function

% SendControl function

```

現在讓我們使用此示例的狀態空間表示形式並設計一個狀態反饋控制器，將保證所需的性能。對於這種情況，我們首先將假定完全進入各州，其次，我們將其放寬為—假設我們只能訪問該職位。就像我們之前所做的我們可以在連續時間或離散時間進行。

先前我們建立了該系統的狀態空間描述，並給出了通過：

$$\dot{x}(t) = \begin{bmatrix} 0 & 1 \\ 0 & -\frac{1}{\tau} \end{bmatrix} x(t) + \begin{bmatrix} 0 \\ \frac{K}{\tau} \end{bmatrix} u(t)$$

其中 $x(t) \in \mathbb{R}^2$ ($x_1(t) = \theta(t)$ 和 $x_2(t) = \dot{\theta}(t)$)， $u(t) \in \mathbb{R}$ (施加電壓)。從穩定時間為 $\%2$ 等於 3τ 的期望性能中，我們得到與以前相同的主導極點，因此特徵方程相同， $\Delta d(s) = s^2 + 2\zeta\omega_n s + \omega_n^2 = 0$ ($\zeta = 0.707$ 和 $\omega_n = 43\zeta\tau$)。使用控制器表示閉環特徵方程式：

$$\det(sI - A + BK) = 0$$

通過均衡這兩個方程式，我們可以得到以下收益：

$$K_1 = 1.1146$$

$$K_2 = 0.0326$$

使用該控制器，輸入幅度等於 30 度表示在圖 11.6 中。從該圖可以看出超調和建立時間就是我們想要的。這很需要注意在穩態狀態下錯誤的存在。這個錯誤可以是如果在循環中添加一個積分動作，則消除。有關此的更多詳細信息，請參閱讀者閱讀[1]。

對於第二種情況，由於無法訪問加載速度，我們可以從該位置計算它或使用觀察者估計系統狀態。因為它前面說過，我們用於觀察者設計的極點應該更快比控制器設計中使用的那些

選擇以下極點 ($s_{1,2}$ ，為設計中實際部分的四倍) 控制器)：

$$\begin{aligned} s_{1,2} &= -4\zeta\omega_n \pm j\omega_n \sqrt{1 - \zeta^2} \\ &= -83.4218 \pm 20.8618j \end{aligned}$$

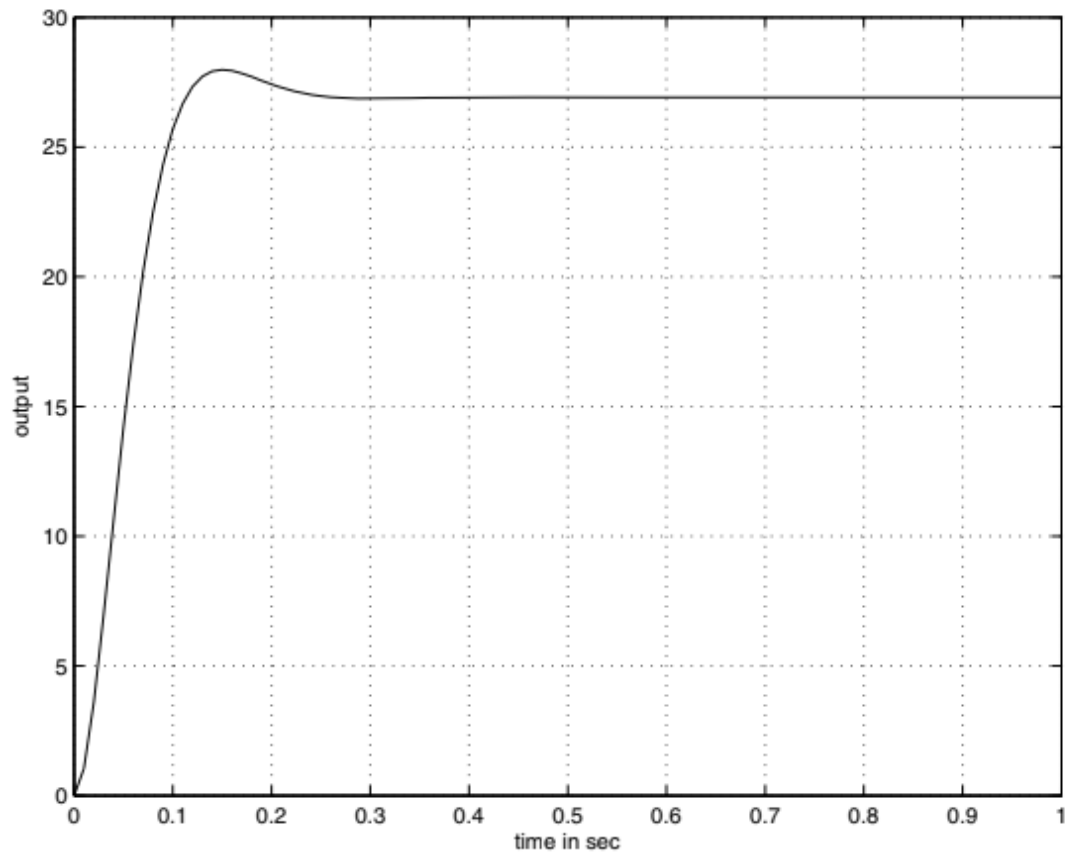


Fig. 11.6 Time response for a step function with 30 degrees as amplitude

我們為觀察者帶來以下收益：

$$L_1 = 151.2$$

$$L_2 = 5029.4$$

控制器的增益與完全訪問控制器的情況相同。狀態向量。

在下面的 Matlab 中，我們提供控制器和觀察器的設計同時給出模擬，以顯示系統和觀察者。

```
clear all
```

```
%data tau=0.064 k=48.9
```

```
A = [0 1;0 -1/tau];
```

```
B = [0 ; k/tau];
```

```
C = [1 0];
```

```
D = 0;
```

```
% controller design
```

```
K = acker(A,B,[-3+3*j -3-3*j]);
```

```
L = acker(A',C',[-12+3*j -12-3*j])';
```



```

% Simulation data Ts = 0.01; x0 = [1 ; 1]; z0 =
[1.1 ; 0.9]; Tf = 2; %final time

%augmented system
Ah = [A          -B*K;
      L*C      A-B*K-L*C];
Bh = zeros(size(Ah,1),1); Ch = [C D*K];
Dh = zeros(size(Ch,1),1); xh0 = [x0 ; z0];
t=0:Ts:Tf; u = zeros(size(t)); m =
ss(Ah,Bh,Ch,Dh);

%simulation
[y,t,x] = lsim(m,u,t,xh0);

%plotting figure; plot(t,y); title('Output');
xlabel('Time in sec') ylabel('Output') grid

figure; plot(t,x(:,1:size(A,1))); title('States of the system');
xlabel('Time in sec') ylabel('System states') grid

figure; plot(t,x(:,size(A,1)+1:end)); title('states of the
observer'); xlabel('Time in sec') ylabel('Observer states') grid

```

Figs (11.7) – (11.9) 給出了輸出，系統狀態和觀察者狀態的說明。

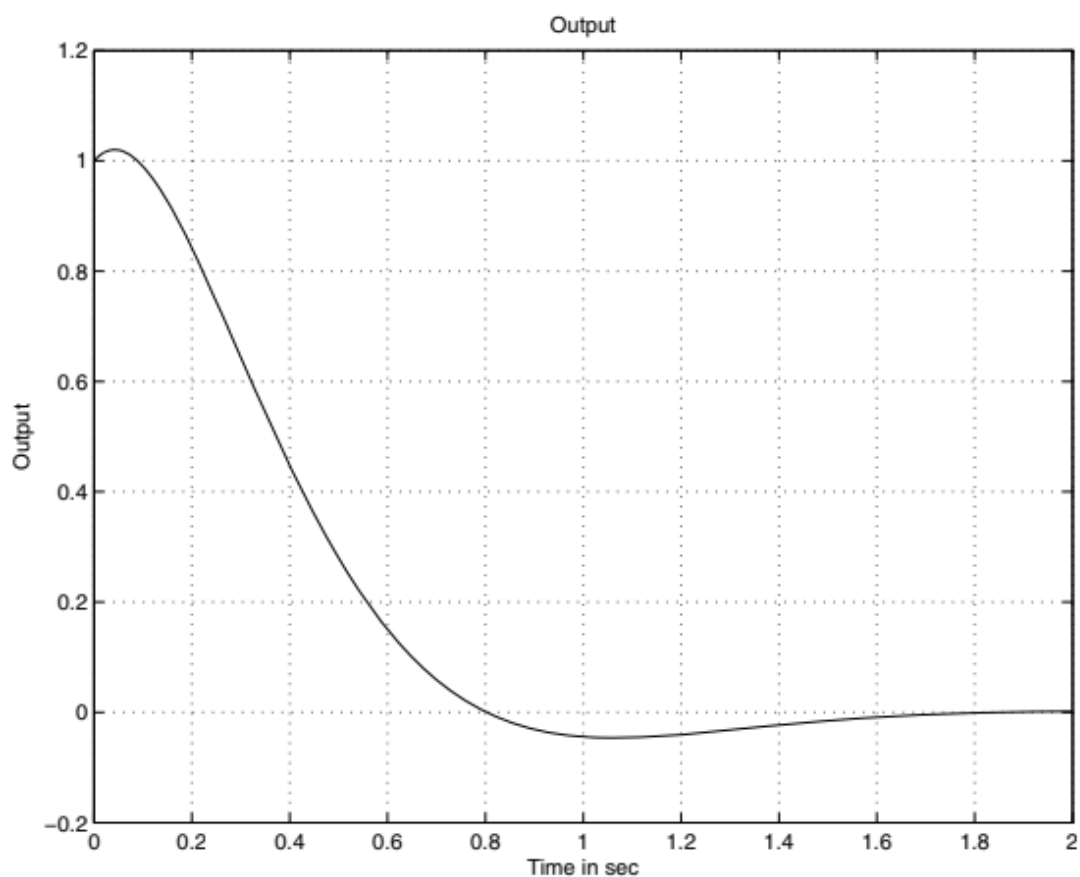


Fig. 11.7 Output versus time

我們還可以使用線性二次方程式設計狀態反饋控制器調節器。實際上，如果我們為成本函數選擇以下矩陣：

$$Q = \begin{bmatrix} 1 & 0 \\ 0 & 5 \end{bmatrix}$$

$$R = 10$$

備註 11.3.2 通常，選擇矩陣沒有魔術規則成本函數。但總的來說，我們使用較高的值來控制實例將強制控件採用較小的值，並可能防止飽和。使用這些矩陣和 Matlab 函數 `lqr`，我們得到：

$$K = 0.3162 \ 0.6875.$$

我們還可以使用魯棒控制的結果設計狀態反饋控制器部分。由於系統沒有不確定性，也沒有外部干擾，我們可以設計一個用於名義動態的狀態反饋控制器。使用系統數據和 Matlab，我們得到：

$$X = \begin{bmatrix} 1.1358 & -0.3758 \\ -0.3758 & 1.1465 \end{bmatrix}$$

$$Y = \begin{bmatrix} -0.0092 & 0.0228 \end{bmatrix}$$

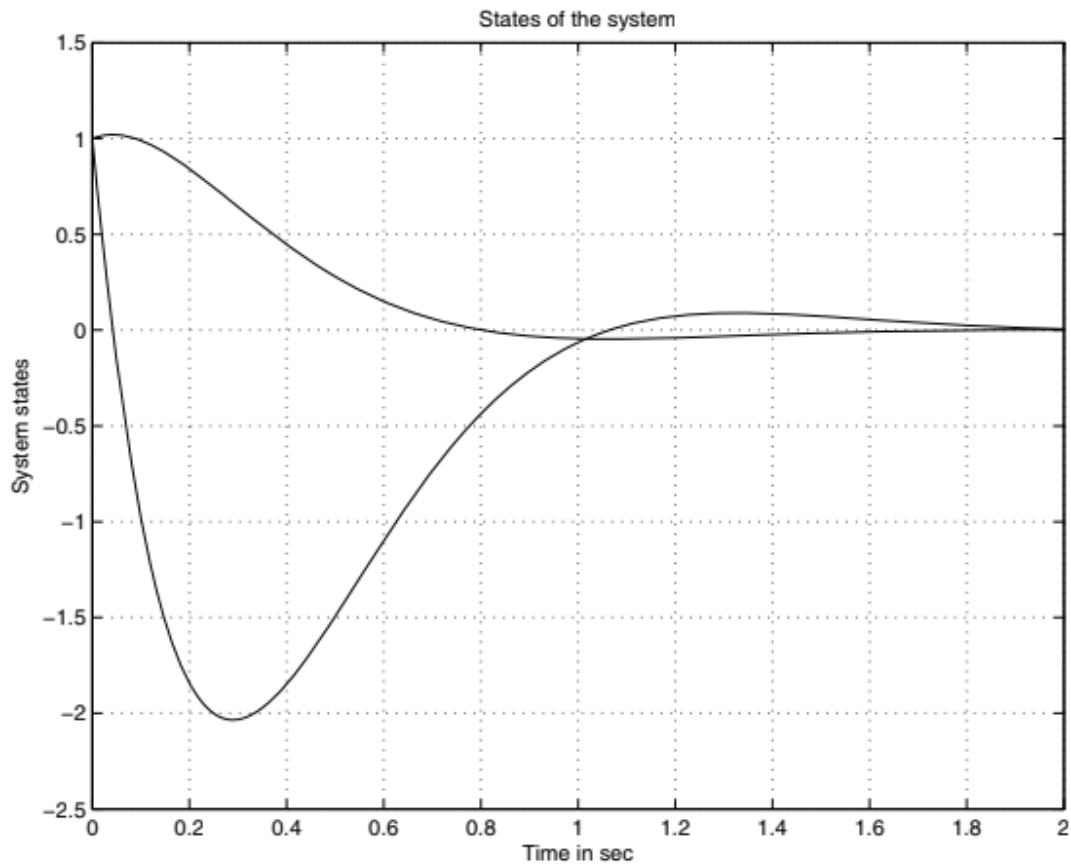


Fig. 11.8 System's states versus time

給出了相應的控制器增益：

$$K = \begin{bmatrix} -0.0017 & 0.0193 \end{bmatrix}$$

備註 11.3.3 由於我們有直流電動機套件的連續時間模型，因此我們用它來設計控制器增益。在這種情況下，我們解決了以下問題

LMI：

$$AX + XA + BY + YB < 0$$

增益 K 由下式得出： $K = YX^{-1}$ 。

有關連續時間情況的更多詳細信息，請向讀者介紹 Boukas [2] 以及其中的參考文獻。

心得：

由於我們設計系的學生大多是純機械科或製圖系進來的，大多沒有機電相關的知識，但是如果要設計機台，就必須要有機電相關的認知，這篇文章能夠教我們如果使用機電系統。