使用模組覆蓋以驗證設計需求

文/ Uttara Kumar,　Nishaat Vasi,
MathWorks 公司

今天是一個大日子。你們團隊正準備要將設計好的系統部署在硬體上，進行下一步的整合測試；然而，最後一刻卻出了個簍子:系統行為異常，它的**速度閥值**和生產程式碼(production code)不符。怎麼會這樣? 團隊明明已經完全遵照標準驗證流程，進行了以下事項：

1. 檢視系統及軟體設計規格需求。
2. .在 Simulink 下模擬演算法。
3. 使用功能測試(functional test)案例來驗證設計。
4. 已針對產生的程式碼進行驗證，來確認程式覆蓋率(code coverage)是足夠的。

那麼，為什麼進行功能測試時無法預測出這種意料外的行為？更重要的，到底要進行多少「測試」才夠？

功能測試的正確性，或者換個說法，單獨驗證模型實現的輸入- 輸出行為，這並不能保證設計的正確性。功能測試基本上從規格需求(requirements)衍生而來，而這些需求可能本來就不完整、不正確、或過度狹隘，因此，想透過功能驗證技術去偵錯變得非常困難，原因是需求本身就存在瑕疵。

而結構式的驗證方式，例如模組覆蓋率(model coverage)，則可指出模型中未執行或未使用的模擬路徑。藉由探測這些未測試過的路徑，你可以偵測出潛在的設計錯誤，並且驗證規格需求。模型覆蓋量測對於必須符合 DO 178C、ISO 26262、以及 IEC61508 標準的應用程式而言，是非常有用的。

本文將簡要說明，如何使用模型覆蓋分析去測試巡航控制器元件的工作流程。本模型包含了一個 PI 控制器，用以計算實際速度和目標速度差之油門輸出 (圖 1) 。

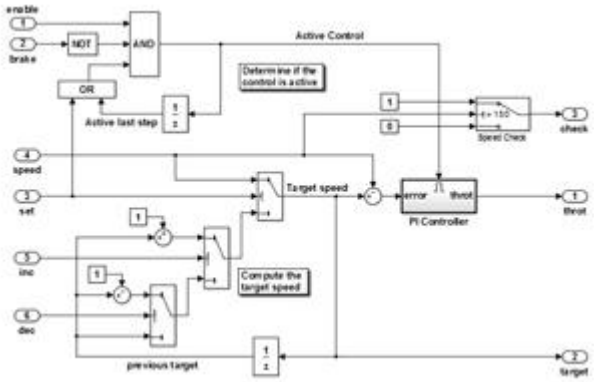


圖 1. 巡航控制系統之控制器設計

設定測試工具

針對控制器，我們使用 Simulink 驗證及有效性檢測模塊組 (Simulink Verification and Validation™) 來建立一個**訓練模型(harness model)** (圖 2)。

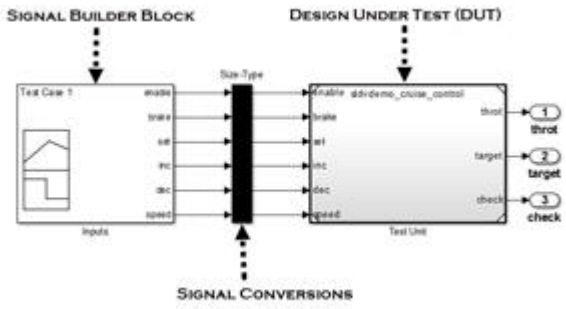


圖 2. 以 Simulink 驗證及有效性檢測模塊組產生之訓練模型元件

在訓練模型裡的訊號產生器 (Signal Builder) 包含了描述我們想測試設計的輸入情境的測試向量。在訊號產生器中，雖然我們可以手動建立這些測試向量，但我們想要重新使用之前我們用來測試功能正確性之規格需求的測試案例。為了這麼做，我們簡單地拉出了包含這些測試案例的 Excel 檔案，並把它們匯入訊號產生器模塊中 (圖 3) 。

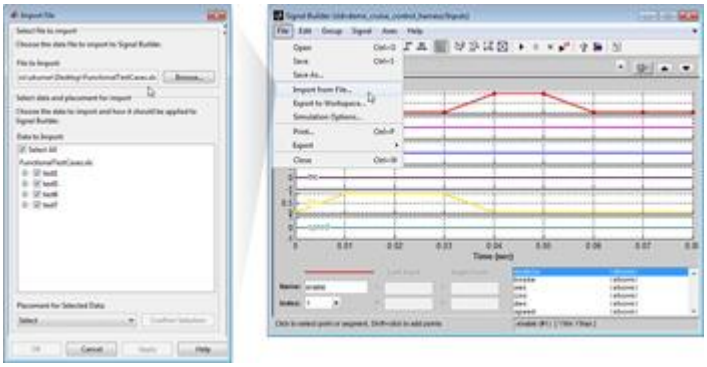


圖 3. 匯入現存測試資料至訊號產生器模塊(Signal Builder block)的介面。每個匯入的訊號群組都代表一個獨特的測試案例。

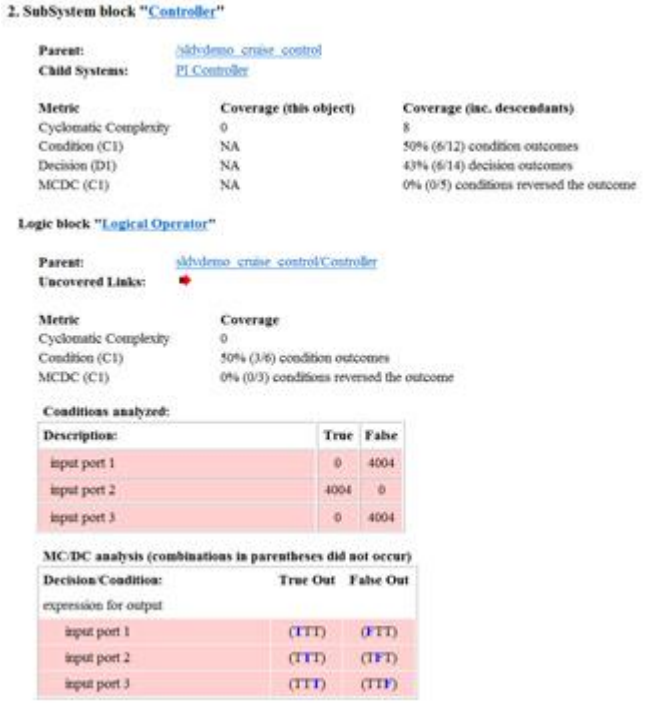
分析模型覆蓋率

然而，這些已測試過設計功能正確性的測試範例，真的能完全的測試我們的設計架構嗎？還有，這些測試範例能履行設計的邏輯路徑到什麼程度？

模型覆蓋分析能幫助我們回答這些問題。本文以巡航控制器為例，我們模擬了待測設計 (DUT)，而全部的測試範例都在訊號產生器模塊中執行。之後，我們將針對各種**覆蓋指標組合(coverage metrics)**，包含條件、決策以及修正條件/決策覆蓋去分析模型。

在模擬的最後，Simulink 驗證及有效性檢測模塊組(Simulink Verification and Validation™) 可產生詳細的**覆蓋指標組合的** HTML 格式報告，內容包含了在待測設計(DUT)中各種不同模型要素。摘要區塊 (Summary) 則能提供整體的**覆蓋指標組合**，而詳細區塊 (Details) 則包含了各個測試要素的**覆蓋指標組合**。

透過報告，我們可以了解哪些設計邏輯路徑在模擬的可能組合中並沒有被測試到(圖 4) 。我們可以使用報告中的超連結去辨認模型中相關的區塊。



2. SubSystem block "Controller"

Parent: sldvdemo_cruise_control
Child Systems: PI Controller

Metric	Coverage (this object)	Coverage (inc. descendants)
Cyclomatic Complexity	0	8
Condition (C1)	NA	50% (6/12) condition outcomes
Decision (D1)	NA	43% (6/14) decision outcomes
MCDC (C1)	NA	0% (0/5) conditions reversed the outcome

Logic block "Logical Operator"

Parent: sldvdemo_cruise_control/Controller
Uncovered Links:

Metric	Coverage
Cyclomatic Complexity	0
Condition (C1)	50% (3/6) condition outcomes
MCDC (C1)	0% (0/3) conditions reversed the outcome

Conditions analyzed:

Description:	True	False
input port 1	0	4004
input port 2	4004	0
input port 3	0	4004

MC/DC analysis (combinations in parentheses did not occur)

Decision/Condition:	True Out	False Out
expression for output		
input port 1	(TTT)	(FTT)
input port 2	(TTT)	(TFT)
input port 3	(TTT)	(TTF)

圖 4. 模型覆蓋分析所產生之 HTML 報告

作為報告的選項之一，我們可以檢視模型內的覆蓋結果。每個模型要素都以顏色代碼標示其覆蓋狀況。綠色代表現有測試已可完全覆蓋或測試，而紅色則代表不完全的結構覆蓋。

在本次的案例中，我們可以清楚看到部分模型要素並未完全覆蓋 (圖 5) 。因此，描述功能正確性的功能覆蓋測試，並未完全測試到其設計。

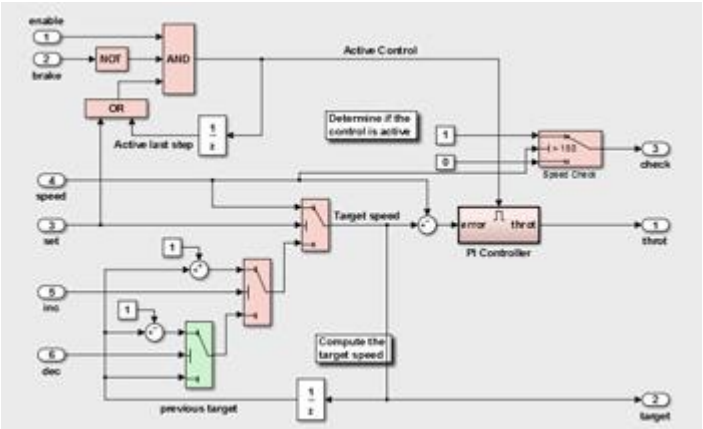


圖 5. 在模型上顯示模型覆蓋結果

我們有兩種選擇去建立測試案例，以處理缺漏的覆蓋。您可透過手動建立測試案例，或是使用 Simulink 設計驗證工具 (Simulink Design Verifier™) 自動生成測試案例。

擴展測試案例以增加模型覆蓋率

在 Simulink 設計驗證工具 (Simulink Design Verifier™) 中，我們可以利用現有的測試案例，針對缺漏的覆蓋去產生測試。我們登入到訓練模型中的現有測試案例中，並且將它們存成 MAT 檔案。之後，我們用 Simulink 設計驗證工具去延伸現有的測試案例 (圖 6) 。

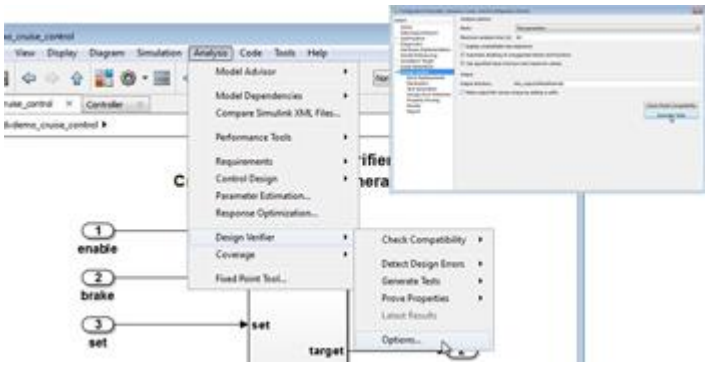


圖 6. Simulink 設計驗證工具中產生測試案例之選項的介面圖

在產生測試案例的最後流程，Simulink 設計驗證工具可產生一個獨立的訓練模型，以及訊號產生器 (Signal Builder) 模塊所產生的測試案例。之後，我們將待測設計 (DUT) 中的所有測試案例和訊號產生器模塊整合在一起 (圖 7) ，然後便可使用這些測試案例去模擬 DUT。

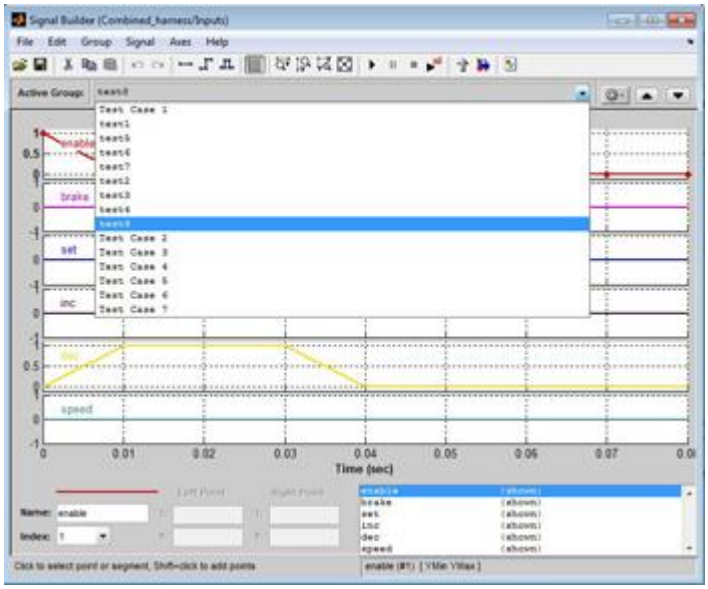


圖 7. 現有的功能測試案例，以及利用 Simulink 設計驗證工具所產生的測試案例

只需要用視覺化的方式來檢查 DUT 的覆蓋結果就可以發現，那些自動生成的測試案例增加了設計結構覆蓋的程度。然而，仍有一個**開關組件(switch block)** 尚未被任何測試所覆蓋 (圖 8)。

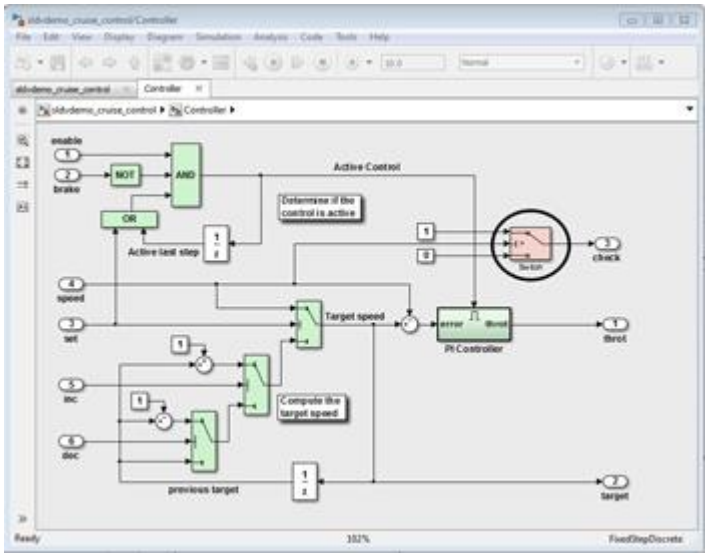


圖 8. 由巡航控制器之模型覆蓋結果得知，仍有一個**開開組件**還有不完全的覆蓋 (紅色區塊)

調查未測試的設計要素。

在測試過程的最後，我們將可以用 Simulink 設計驗證工具產製的報告進行調查。這份報告將說明哪一項測試滿足哪一項覆蓋目標。我們看到與**開關組件**有關的目標，是呈現尚未執行的紅色 (圖 9)。而這個**開關組件**在此組件近似於板機的功能，它會以第二輸入端的值為基礎，選擇通過第一或是第三輸入端。這份報告也指出了，沒有任何一項測試案例可以讓此設計的**開關組件**經過第一輸入端，又是什麼原因造成如此結果呢？

Objectives Proven Unsatisfiable

Simulink Design Verifier proved that there does not exist any test case exercising these test objectives. This often indicates the presence of dead-logic in the model. Other possible reasons can be inactive blocks in the model due to parameter configuration or test constraints such as given using Test Condition blocks. In rare cases, the approximations performed by Simulink Design Verifier can make objectives impossible to achieve.

#	Type	Model Item	Description	Analysis Time (sec)	Test Case
36	Decision	Controller/Speed Check	trigger > threshold true (output is from 1st input port)	1	n/a

圖 9. Simulink 設計驗證工具提供測試報告，顯示此測試目標是尚未執行的。
Figure 9. Simulink Design Verifier report for test generation showing objectives proven unsatisfiable.

藉由 Simulink 設計驗證工具的規格需求追蹤功能 (requirements traceability)，我們可以追溯相關的規格需求，以了解**開關組件**設計的背後動機 (圖 10)。

由此，我們得以了解，**開關組件**要經過第一輸入端的條件是，當第二輸入端的速度超過 150m/s。然而，鑒於我們系統的動力學，以及在其他設計需求上，我們設計的速限只於 0 至 100m/s 之間。因此，速度永遠不會超過 150m/s 這個門檻。

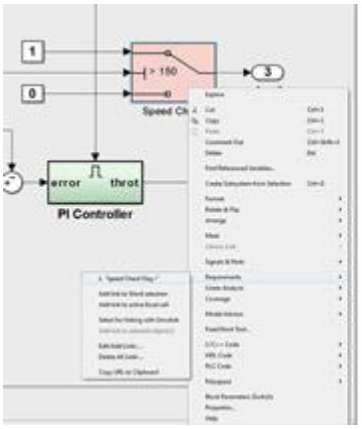


圖 10. 左：Simulink 模型內的**開關組件**　右：與**開關組件**相關的規格需求說明

下一步

在與團隊重新評估系統需求後，我們修正了**速度閥值**，從 150m/s 改為 75m/s，以更佳地表現我們的系統設計。透過修正錯誤、依序更改控制器模型，讓我們得以達成模型覆蓋率 100%的目標 (圖 11) 。現在，這個已經過等價及回歸測試的結構驗證過程，並且獲得程式碼**覆蓋指標**的測試向量，就可以讓我們重新使用了。

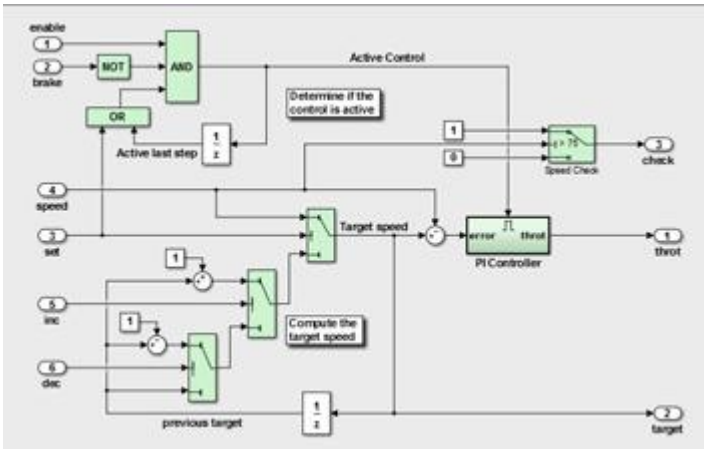


圖 11.模型上顯示 100% 模型覆蓋結果

概括來說，模型覆蓋分析的結構性驗證有助於揭露設計規格需求的問題，尤其是那些能通過功能性驗證的潛在缺失，在這次的案例中，我們在相對簡單的控制器設計上，使用了模型覆蓋並自動產製測試案例。若是對於較大型的設計、具備較多的輸入端及流程圖邏輯較複雜的設計，可能的平行模擬路徑和互動數量就會較高。因其設計複雜度高，需手動建立測試案例以履行所有路徑的機會便較少；因此，結構覆蓋測試就更加地重要。只有在評估現有架構之後，設計架構才得以改善。而諸如模型覆蓋等先期的驗證方法，將有助於提供非常關鍵性的設計評估。