應用串接方法於連續變化轉速之四行程引擎聲音合成

Concatenation-based Method for the Synthesis of Engine Noise with Continuously Varying Speed

吳銘冠 Ming-Kuan Wu　　陳嘉平 Chia-Ping Chen
國立中山大學資訊工程系
Department of Computer Science and Engineering
National Sun Yat-Sen University
M003040056@student.nsysu.edu.tw, cpchen@cse.nsysu.edu.tw

摘要

在本研究中，我們提出並實做一個串接式聲音合成系統，合成的標的物件是連續變化轉速之引擎聲音。我們提供一個繪圖的介面讓使用者畫出連續變化的引擎轉速曲線作為系統的輸入，然後輸出對應的引擎噪音。採用繪圖的方式，不僅能讓輸入更有彈性，也能減少輸入所需要的時間。主觀測試的實驗結果顯示，合成出來的聲音在自然度的測試上以及和原始引擎聲的相似度比較上有良好的表現。本論文所提出的方法，可以推廣到其他物理產生過程機制清楚簡單的聲音物件。此外，也可以應用到虛擬實境訓練或遊戲等等。

關鍵詞： 聲音物件合成、串接合成方法、引擎噪音合成、虛擬實境

Abstract

In this study, we propose and implement a concatenation-based audio signal synthesis system for the engine noises of continuously varying speed. A user simply draws the engine speed curve through an interface, and the corresponding audio signal is synthesized as output. This drawable interface makes the input function flexible and reduces the input time. The implemented system was evaluated with subjective tests. Overall, the performance was good regarding quality and similarity. The proposed method can be feasibly applied to the synthesis of any sound objects which are produced with a clear and simple physical process. Furthermore, the technology can be integrated to virtual reality, such as in training and gaming applications.

keywords: audio object synthesis, concatenation synthesis method, engine noise synthesis, virtual reality

一、緒論

(一)、研究背景、動機

聲音合成技術在人機介面裡扮演著重要的角色，目的是將聲音用人為的方式產生，其中串接式合成方式為主要的合成技術之一。此合成方法是從錄製的聲音中找出所需的合成單元，接著再做一些韻律方面的處理，之後將聲音單元串接。通常使用此方法得到的聲音自然度和品質都相當不錯。在虛擬實境(Virtual Reality, VR)的機車引擎聲或是坊間的賽車遊戲，往往用到的引擎聲都是預先錄製好的 [1]，這些錄製好的音檔，雖然品質較佳，但在錄製時往往需要大量的時間和人力，且缺乏彈性。因此在這裡提出一個手動繪圖的合成方式，來簡化輸入合成資訊的步驟，以四行程檔車的引擎聲為例，利用最短時間和最少資源，來合成上述應用程式所需要的音檔。

(二)、相關研究

1、聲音合成

在聲音合成技術裡，基週同步疊加法(Pitch Synchronous Overlap Add, PSOLA) [2]為串接式合成常用的調整動作。此方法先將波形分解成許多的基本波形，再將基本波形疊加以得到合成的聲音波形。關於基本頻率和音長的調整，可利用基本波形的重疊間隔和數目來達到，為現在常見的合成方法之一。但此方法的缺點為，在相鄰的合成單元的串接邊界上，若建立合成單元庫時採用自動作切割的話，可能會造成共振峰軌跡銜接不平順，降低合成聲音的流暢度。

除了PSOLA 的方式之外，還有語料庫為主(Corpus-based)的合成方式 [3]。其方法為先錄製大量的語料，然後在合成時根據演算法從許多候選單元中選出一組會讓合成音最為自然的組合。由於合成單元的選擇法並不會對錄製的語音作太多的信號處理動作，此外可供候選的合成單元數目很多，使得語音單元間的不連續被降低很多，因此合成音的自然度上是相當不錯的。在本文，我們簡化串接式語料庫為主的合成方式，改以引擎聲音來當作合成單元，因此可以原音重現，具有極佳的合成音質，進而合成出特定範圍的引擎聲。近年來，上述串接合成方式已應用在不少系統中且都有不錯的表現，如微軟亞洲公司之木蘭(MULAN)系統 [4]和訊飛中文語音系統。

2、引擎合成

在國外，諧波同步疊加法(Harmonic Synchronous Overlap and Add, HSOLA) [5]被使用來合成引擎噪音。此篇論文提到先採樣一個不斷變化預錄的引擎聲，然後使用諧波同步累加法的方式。該方法的目的主要是減少階段式的不連續性，使其聽起來更具有連續性。合成信號的和諧性被保留，提高了恢復原狀的音質。在其他的研究中發現到，車輛產生的聲波波形，是由兩個部份的總和所組成 [6]。第一個是由引擎旋轉部件所產生諧波相關的一連串音調，而第二個是由輪胎摩擦所產生的噪音。但在本文的引擎噪音合成裡，為了減少合成的複雜度，故不考慮輪胎摩擦所產生的噪音。

(三)、系統概述及研究方向

本文的研究重點是嘗試以繪圖的方式輸入所需要的資訊，希望能減少輸入資訊所需要的時間。也希望能更有彈性的，在特定轉速範圍間，能夠合成出想要的轉速音檔，本文中的轉速皆以每一分鐘的轉速(rpm)為單位。在此篇論文中，因為採用串接的方

式，合成出來的聲音在音色的自然度上有不錯的表現。圖1為系統概述圖，一開始可以選擇兩種使用者介面來輸入所需要的資訊，分別是以文字的方式或是以繪圖的方式輸入資訊。文字輸入的資訊包括開始時轉速、結束時轉速和合成時間。繪圖輸入的資訊包括合成時間以及繪圖的曲線。採用繪圖輸入資訊的方式能更有彈性且快速的產生欲合成的音檔。

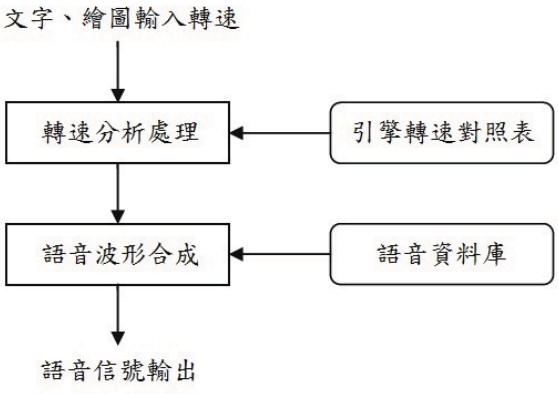

圖 1、 輸入轉速資訊和信號輸出系統架構圖

(四)、四行程引擎簡介

四行程引擎(Four Stoke Engine)完成一次循環，必須經過「吸入、壓縮、點火、排氣」四個步驟 [7]，其運作的程序分別是：

◊ 吸入行程：活塞往下，進氣閥打開，將空氣與燃料的混合氣吸入汽缸中。
◊ 壓縮行程：進氣閥關閉且活塞往上，壓縮此混合氣使體積變小。
◊ 點火行程：在壓縮的混合氣中點火，使氣體燃燒爆發並推動活塞往外作用。
◊ 排氣行程：此時排氣閥打開且活塞再度往上，將燃燒後之廢氣排出汽缸。

根據以上四個行程，可以發現到當完成一個循環時，引擎轉了兩次。

二、合成單元收集

由於引擎聲的轉速在時域上主要為遞增或遞減的連續性變化，故在錄製音檔時，盡可能的收錄大量的連續遞增或遞減音檔。在這一節裡主要是說明音檔的錄製、分析和合成單元產生的過程。

(一)、音檔錄製

本文所收錄的音檔為野狼125 檔車的引擎聲，音檔共分為兩個部份。第一個部份為一個長達3 分鐘左右遞增的引擎轉速音檔，將它令為*SetA* ；第二個部份為評測時所需要合成的測試音檔，將它令為*SetB* 。*SetA* 錄製的方式為，採用人為的方式來線性增加油閥的大小，以達到線性成長的轉速。但由於是以人為的方式來增加轉速，故很難達到線性增加轉速，所以合成單元無法依照線性的時間來做切割，故我們將在之後的章節來解決這個問題。*SetB* 為2 到16 秒共10 個不同轉速範圍的音檔，且轉速的變化為人為隨機產生。轉速的範圍介於1000 轉到3000 轉之間，其轉速變化與時間資訊如表1 所示。

編號	轉速範圍	秒數範圍
1	1000-2700	(0)-(16)
2	1160-2990-1530-2379-1250	(0)-(1.8)-(2.7)-(4.1)-(6.6)
3	1235-2783-1585	(0)-(3.4)-(8.8)
4	1454-1520-2961-2259	(0)-(3)-(3.8)-(4.1)
5	1030-2852-1113-2213-1208 -2786-1123-2570-1213-2790 -1206-2208-1310-2785-1630	(0)-(0.2)-(0.8)-(1.3)-(1.5)-(2.2) -(2.7)-(3.1)-(3.4)-(3.9)-(4.5) -(4.8)-(5.1)-(5.6)-(6)
6	1651-2772-1498	(0)-(1.6)-(5.8)
7	1635-1635-2901-1954	(0)-(1.5)-(1.9)-(2.6)
8	1628-1736-2493-1978	(0)-(2.3)-(3.5)-(4.3)
9	1972-1972	(0)-(2.1)
10	1111-2706	(0)-(2.7)

表 1、 *SetB* 音檔概要資訊

(二)、音檔分析

若將引擎的聲音以waveform 的形式表示，會發現到聲音的變換是非常具有規律性的。將此音檔改以在頻譜上顯示，更容易發現其規律性的變化，因此我們著重於頻譜的部份。圖2 為*SetA* 音檔其中一段引擎聲音的片段，所產生的waveform 和所對映的頻譜圖。

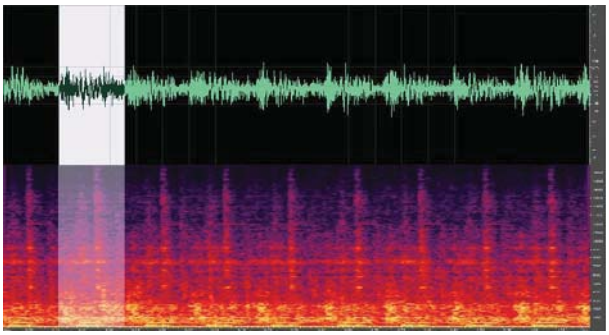

圖 2、 上半部為SetA 其中一片段的waveform ，下半部為其對映的頻譜圖。

根據之前四行程的引擎運作原理，我們發現到完成一次循環，引擎共轉了兩次。且此一循環也是引擎聲變換的一個週期，故我們可以根據此訊息來計算引擎的轉速。也就是說我們只需要計算一個週期當下的sample 數，就可以得知其當下的轉速，轉速的計算公式如下：

$$\text{cycle per minute} = \frac{\text{sample rate}}{\text{sample in the cycle}} * 2 * 60, \quad (1)$$

其中，在本文裡的sample rate 為44100Hz ，cyclesamples 為一個合成單元的sample 數，cycle per minute 為此合成單元每一分鐘的轉速。

(三)、合成單元的產生

根據轉速計算公式，找出*SetA* 音檔1000 轉到3000 轉的範圍，並以overlap 的方式切成2000 個一秒左右的片段。但為了方便起見，我們將其編號為1000 至3000 並且只選取以10 為單位的編號，共201 個片段。

接著將這些片段做頻譜的擷取來分析其頻率，如圖3(b) 所示。根據matlab 頻譜圖的色度表，能量大到能量小顏色的變化為紅色到藍色，其中引擎聲的能量都集中於黃色和紅色。黃色的色度值為-25 ，故我們將色度大於-25 的部份設為1 ，小於-25 部份設為0 。然後將縱軸上的值累加起來，重新產生一個根據能量分佈的曲線圖，如圖3(c) 所示。

之後，再根據此圖以人為的方式找出橫軸的切值。判斷的規則分別為要能切出最多週期，並且要能接近最大峰值。將大於此值以上的部份保留，小於此值的部份設為0 。並重新繪製出多個錐狀的圖，如圖3(d) 所示。

接著將每個錐狀體一開始非零的部分標記起來，最後將相鄰錐狀體標記的值相減，就可以得出此一編號多個合成單元。

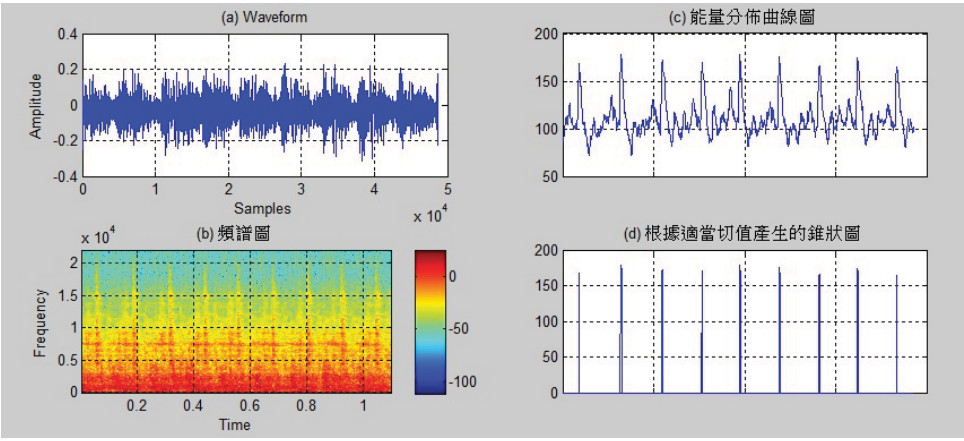

圖 3、 編號1000 的音檔片段所產生的waveform(a)，頻譜圖(b)，經由色度表重繪的能量分布曲線圖(c)，根據適當切值重新繪製的錐狀圖(d)。

經由以上的方法共切出2015 個合成單元。但根據轉速計算公式，因為重複的關係，只產生260 個不同轉速的合成單元。令其轉速為$U = \{u_i | i = 1, ..., 260\}$ 。接者，我們令V為欲找的轉速，如下式所示：

$$V = \{v_j | 1000 + (j-1) * 10, \quad j = 1, ..., 201\}, \tag{2}$$

之後再根據$|v_j - U|, \quad j = 1, ..., 201$ 取差值最小的u_i 來代替v_j 。部分對映如表2 所示。且其轉速與sample 數的關係為近似一個如圖4 的反曲線。

表 2、 編號1 至編號10 的轉速對照表

編號	欲找轉速(V)	近似轉速(U)	編號	欲找轉速(V)	近似轉速(U)	...
1	1000	1003	6	1050	1048	...
2	1010	1008	7	1060	1058	...
3	1020	1022	8	1070	1069	...
4	1030	1032	9	1080	1080	...
5	1040	1042	10	1090	1091	...

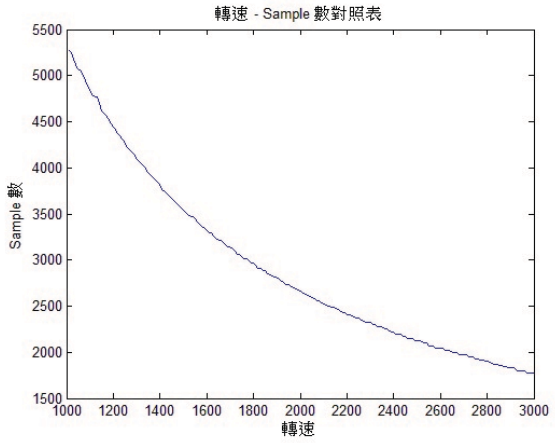

圖 4、 轉速與sample數關係圖。

三、系統架構

(一)、環境及介面

本文的引擎聲合成系統建構在matlab 環境中，其中有兩個使用者介面。第一個使用者介面為文字輸入介面，可以產生遞增或是遞減的合成引擎聲，如圖5 所示。第二個使用者介面為繪圖合成介面。當輸入完所要產生音訊的秒數時，會自動產生一個畫布，以供使用者來繪製引擎的轉速資訊。其中轉速的範圍介於1000 轉到3000 轉之間，如圖6 所示。

(二)、合成方式

在文字輸入介面，根據使用者輸入的開始轉速、結束轉速和時間來獲得合成所需要的資訊，接著我們將對應的轉速合成單元平均分配到適當的轉速範圍，分配方式如下：

◊ 若在時間內轉速變化大的話，則平均適當的挑選合成單元；
◊ 若在時間內變化小的話，則平均適當的重複挑選所需的合成單元；
◊ 開始轉速 ＜ 結束轉速則為遞增，帶入遞增演算法；
◊ 開始轉速 ＞ 結束轉速則為遞減，反向的帶入遞增演算法；
之後將所有的轉速單元串接起來獲得一個新的合成音檔。

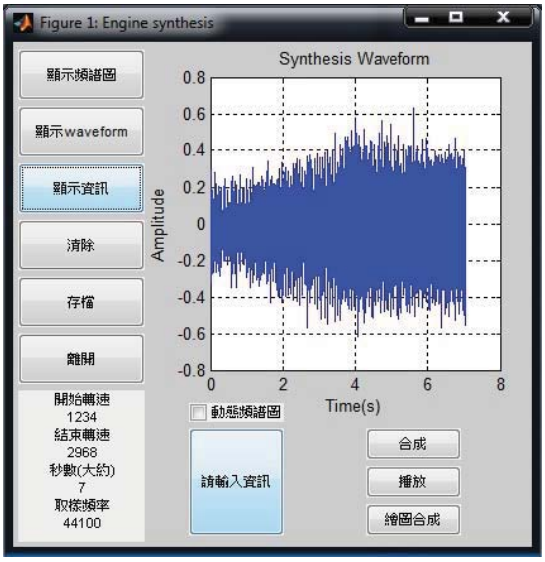

圖 5、 文字輸入介面

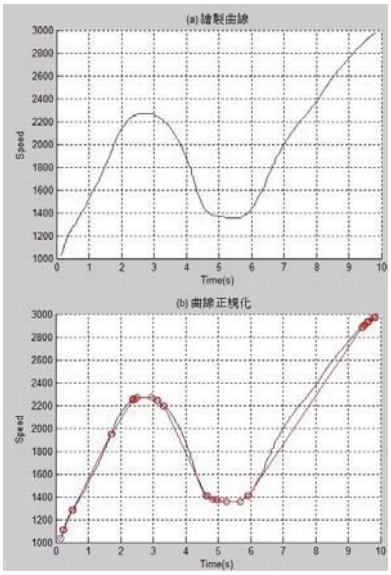

圖 6、 上半部為使用者繪製的曲線(a)，下半部將此曲線標示mark 並正規化(b)

在繪圖介面，使用者先輸入欲合成的時間資訊t 。之後會產生縱軸為轉速，橫軸為時間的繪圖介面，如圖6(a) 所示。我們令橫軸為t ，縱軸為y 。當使用者繪製完轉速曲線時，此時系統會根據以下的演算法將曲線正規化，如圖6(b) 所示。

- $t(start)$ 和$t(end)$ 標示為mark；

- 找出轉折點：
  - ◊ 若$t(i) > t(i-1)$ 且$t(i) > t(i+1)$ ，將$t(i)$ 標示為mark ；
  - ◊ 若$t(i) < t(i-1)$ 且$t(i) < t(i+1)$ ，將$t(i)$ 標示為mark ；
  - ◊ 若$t(i) > t(i-1)$ 且$t(i) = t(i+1)$ ，將$t(i)$ 標示為mark ；
  - ◊ 若$t(i) = t(i-1)$ 且$t(i) > t(i+1)$ ，將$t(i)$ 標示為mark ；
  - ◊ 若$t(i) < t(i-1)$ 且$t(i) = t(i+1)$ ，將$t(i)$ 標示為mark ；
  - ◊ 若$t(i) = t(i-1)$ 且$t(i) < t(i+1)$ ，將$t(i)$ 標示為mark ；

- 將相鄰的mark連接起來，產生欲合成的多個片段；

- 將所有片段根據文字合成的演算法串接成一個輸出音訊。

一、實驗與評測

在評測的部分主要分為聲音的自然度測試，和原始音檔的相似度測試，受測人數為10 人。在自然度測試中，根據MOS的5分評分制，每位受測者在聽完每句合成的音

檔之後，隨即在聲音品質上的表現給予1 到5 分的分數。在相似度測試中，評分的規則也類似MOS 的5 分制度。但將其改成相似度的比較，評分標準如表3 所示：

分數	品質	註解	分數	品質	註解
5	優秀	聲音相當自然	5	優秀	聲音相當相似
4	很好	聲音自然	4	很好	聲音相似
3	普通	聲音品質可以接受	3	普通	聲音相似度可以接受
2	不好	聲音不自然	2	不好	聲音不相似
1	糟糕	聲音非常不自然	1	糟糕	聲音非常不相似

表 3、 左半部表格為MOS主觀評測標準表，右半部表格為相似評測標準表。

(一)、聲音自然度測試

在聲音的自然度測試上，我們根據曲線繪圖介面隨機產生8 個音檔。其時間為2 到10 秒不等，以便用來做聲音的自然度測試。8 個繪製曲線如下分類：

◊ 2 秒音檔：低轉-高轉、高轉-低轉，共兩個音檔。
◊ 5 秒音檔：低轉-高轉-低轉、高轉-低轉-高轉、低轉-低轉、高轉-高轉，共四個音檔。
◊ 10 秒音檔：多個上下起伏的轉速，共兩個音檔。

以上8 個音檔的曲線繪製和其編號如圖7 所示。

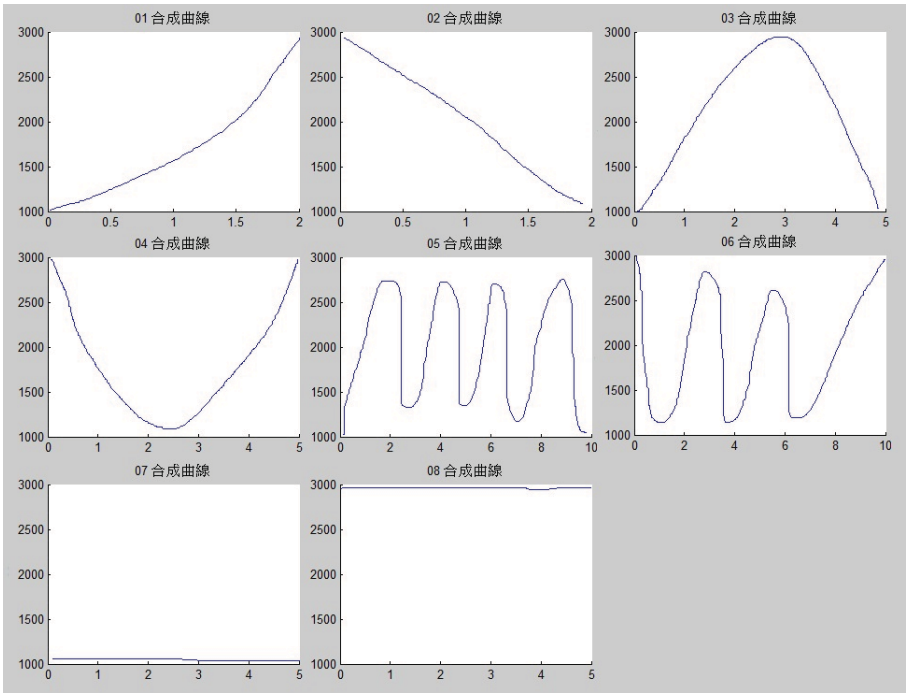

圖 7、 圖中為依序編號的曲線圖。橫軸為秒數，縱軸為引擎轉速。

(二)、聲音相似度測試

將*SetB* 裡的10 個音檔做時間上轉速概要的分析，其資訊如表1 所示。根據這些測試音檔的資訊來產生合成的引擎聲，接著和原始的音檔做比較並且評分。

(三)、實驗結果

在自然度的測試上，我們可以發現到普遍都表現不錯，如表4 所示。但是編號7 和8 的音檔分數明顯的低落。分析其原因為，音檔7 為繪製低轉速的水平直線，音檔8 為繪製高轉速的水平直線，這將導致不明顯的轉速變化，進而使得合成的品質較為不好。在相似度的測試上，我們可以發現到分數也是不錯的，如表5 所示。但是編號7 和9 的音檔分數明顯的低落。分析其原因為，編號7 音檔前面部分的轉速變化較不明顯；編號9 音檔的轉速變化也不明顯，因而導致合成出來的品質較為不好。

編號	1	2	3	4	5	6	7	8
分數	4.7	4.3	4.1	4	3.5	4	2.8	2.7

表 4、 自然度評分

編號	1	2	3	4	5	6	7	8	9	10
分數	4	4.3	3.9	3.5	3.7	4.1	3	3.7	2.3	4.5

表 5、 相似度評分

五、結論與未來方向

本系統為基於串接式合成的引擎聲合成系統，並根據引擎轉速，且採用繪圖的方式來產生合成所需要的資訊。使用本系統能更有彈性的輸入資訊，且更能加快輸入資訊所需要的時間。在主觀實驗中，合成的聲音在自然度和原始音檔的相似度上，是令人滿意的。使用此合成系統，不只可以應用在引擎聲的合成，也可應用在在物理產生過程較為簡單的物件，例如雨聲、燒開水聲、海浪聲，甚至鼓聲等等。本系統在實作上也有幾個缺點，雖然串接式合成能有較佳的品質，但在音訊參數的調適上彈性較差。另外，使用本系統，在長時間相同轉速或者轉速變化較少的合成音檔裡，主觀評測的分數明顯較差。原因為，串接合成單元間的變化很小，導致音檔聽起來較不真實，這也是未來要克服的問題之一。在未來的方向裡，為了使合成單元能夠更為準確，在合成單元的產生部分，也可使用pitch mark 來偵測，以找出較準確的合成單元。另外，我們也可以將油門把手的資訊繪製成轉速曲線再進行合成，也就是說可以直接轉動把手來合成出想要的引擎聲，這些都是在可行的應用範圍之內。

參考文獻

[1] Carsoop, “Forza Motorsport 4: Heres how they record car engine sounds,” introduction: http://carscoop.blogspot.com/2011/06/forza-motorsport-4-heres-how-they.html, 2011.

[2] E. Moulines and F. Charpentier, “Pitch-synchronous waveform processing techniques for text-to-speech synthesis using diphones,” *Speech Commun.*, vol. 9, no. 5-6, pp. 453–467, Dec. 1990.

[3] B. Ao, C. Shih, and R. Sproat, “A corpus-based Mandarin text-to-speech synthesizer,” in *ICSLP’94*, 1994, pp. –1–1.

[4] M. Chu, H. Peng, Y. Zhao, Z. Niu, and E. Chang, “Microsoft mulan - a bilingual tts system,” in *ICASSP 2003*, pp. 264–267.

[5] J. Jagla, J. Maillard, and N. Martin, “Sample-based engine noise synthesis using a harmonic synchronous overlap-and-add method,” in *Proceedings of ICASSP 2012*, Kyoto, Japan, Mar. 2012, p. poster.

[6] Y. Ban, H. Banno, K. Takeda, and F. Itakura, “Synthesis of car noise based on a composition of engine noise and friction noise.” in *ICASSP*. IEEE, 2002, pp. 2105–2108.

[7] V. Chen, “4-stroke engine,” introduction: http://www.bizol.com.tw/video.aspx?cid=12, 2011.