บทที่ 4

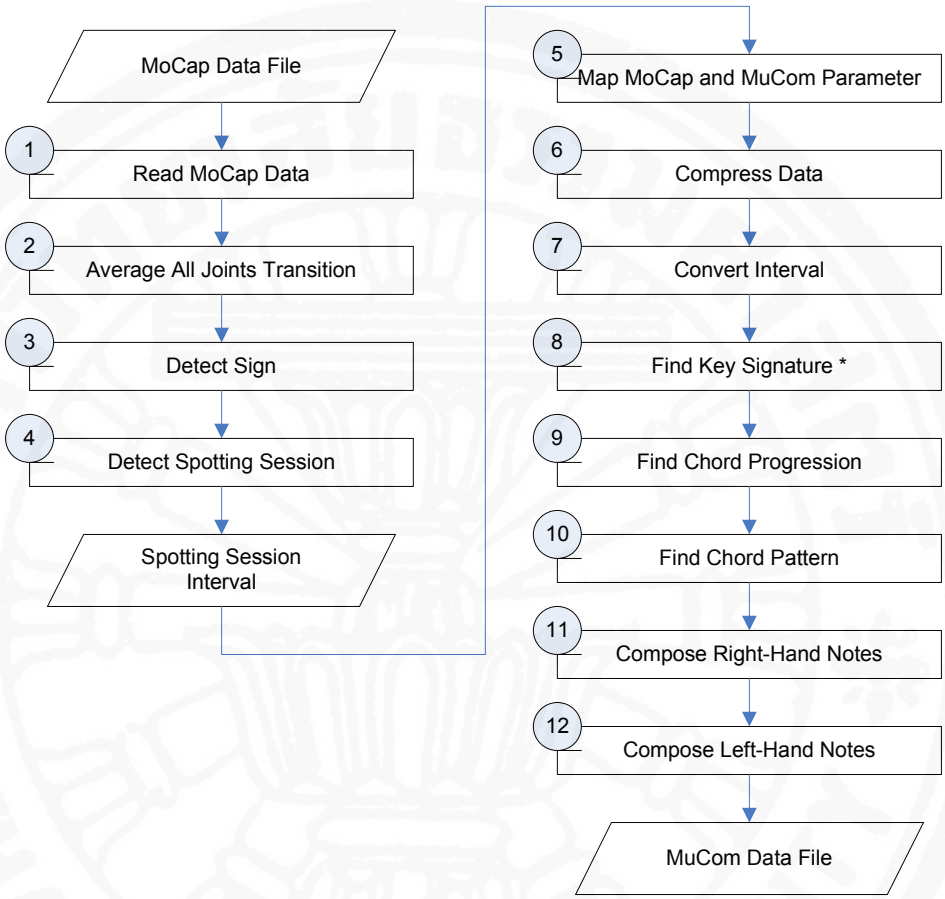
การสร้างระบบสำหรับวิทยานิพนธ์

หลักการที่ใช้ในการวิจัย Natural Music Composition นั้นประกอบด้วยวัตถุดิบที่ได้มาจากธรรมชาติซึ่งในที่นี้หมายถึงข้อมูลที่เก็บมาจากเครื่องจับการเคลื่อนไหวของมนุษย์ (Human Motion Capture Data) นั่นเองที่นำมาประมวลผลในกระบวนการของการประพันธ์ดนตรี (Music Composition) จากหลักการที่ว่านี้สามารถทำให้ดนตรีที่เกิดขึ้นนั้นมีความเป็นธรรมชาติมากกว่าเมื่อเทียบกับงานวิจัยที่เกี่ยวข้องในด้านการประพันธ์ดนตรีเป็นหลัก รวมทั้งการที่ได้ผนวกวิทยานิพนธ์ชิ้นนี้เข้ากับทฤษฎีทางดนตรี โดยนำเข้ามากำกับผลลัพท์จากการประมวลผลให้อยู่ในแบบแผนตามทฤษฎียิ่งเสริมความไพเราะและเป็นดนตรีมากกว่างานวิจัยชิ้นอื่นๆ

ขอกล่าวถึง Motion Capture และ Music Composition ซึ่งเป็นอินพุทและเอาท์พุท ในวิทยานิพนธ์นี้ว่า MoCap และ MuCom ตามลำดับ

ภาพที่ 4.1

ขั้นตอนการทำงานของระบบ



ในการทำงานของระบบนั้นดำเนินขั้นตอนดังรูปที่ ขั้นตอนหลักๆของวิทยานิพนธ์ชิ้นนี้ได้ออกแบบไว้เป็น 3 หัวข้อ โดยเริ่มจาก 1. ขั้นตอนการกรองข้อมูลจากการเคลื่อนไหว (Motion Capture Data-Filtering), 2. ขั้นตอนการเชื่อมต่อระหว่างการเคลื่อนไหวและดนตรี (Motion and Music -Synchronization) และ 3. ขั้นตอนการประพันธ์ดนตรี (Music Composition) โดยขั้นตอนในการสร้างระบบขึ้นมานั้นจึงได้ลงในรายละเอียดของฟังก์ชั่นการทำงานดังที่แสดงอยู่ในภาพที่ 4.1 ซึ่งในทางฝั่งซ้ายแสดงส่วนของขั้นตอนการกรองข้อมูลจากการเคลื่อนไหว และฝั่งขวาแสดงขั้นตอนการประพันธ์ดนตรี โดยส่วนของการเชื่อมต่อระหว่างการเคลื่อนไหวและดนตรีนั้นได้กำหนดไว้ในส่วนต้นของขั้นตอนการประพันธ์ดนตรี

ขั้นตอนการเตรียมการก่อนเรียกใช้ระบบ (Pre-Processing)

- ในการสร้างระบบขึ้นมานั้นได้เขียนโค้ดด้วยภาษา Java โดยพัฒนาบน Eclipse 3.2 และใช้ JRE 1.5.0.11 ในส่วนของการสร้างเอาท์พุทที่เป็นรูปแบบ MIDI นั้นได้ใช้ Library ของ JMUSIC 1.5 หาได้จาก [W073] โดยเครื่องมือทุกอย่างนั้นเป็น Open Source ทั้งหมด
- แอปพลิเคชั่น Maya ที่ใช้นั้นมีเวอร์ชั่นที่เป็น Learning Edition ที่สามารถทดลองใช้ได้ด้วย และแอปพลิเคชั่นที่ใช้ในการแสดงโน้ตในรูปแบบของบรรทัด 5 เส้นนั้นได้แก่ Cakewalk Pro Audio
- เริ่มต้นด้วยการเตรียมข้อมูลของการจับความเคลื่อนไหวของมนุษย์ในหลายๆรูปแบบของการเคลื่อนไหวไว้เพื่อทดสอบการยอมรับของระบบ สำหรับรูปแบบไฟล์ของ MoCap นั้นมีหลายหลายรูปแบบเช่น Biovision (BVH) หรือ Acclaim (AMC, ASF) เป็นต้นซึ่งมีวิธีจัดเก็บข้อมูลแตกต่างกันไป แต่วิทยานิพนธ์นี้ได้เลือกใช้รูปแบบไฟล์ BVH โดยดาวน์โหลดมาจาก [W071] โดยรูปแบบไฟล์นี้ได้รวมไว้ทั้งลำดับตำแหน่งของข้อต่อต่างๆรวมถึงการเคลื่อนไหวในแต่ละช่วงเวลาซึ่งเพียงพอที่จะนำมาใช้ในวิทยานิพนธ์นี้
- ในการเรียกดูการเคลื่อนไหวที่ได้บันทึกไว้ในไฟล์ BVH นั้น สามารถเปิดดูได้จากแอปพลิเคชั่น Maya ซึ่งมีการใช้อย่างแพร่หลายและสามารถสร้างงานที่เกี่ยวข้องกับ 3D ได้อย่างมีประสิทธิภาพ การทำงานที่เรียกใช้นั้นเป็นเพียงส่วนน้อยของแอปพลิเคชั่นเท่านั้น โดยใช้ MEL Script ซึ่งเป็นการใช้งานการเขียนโปรแกรมสำหรับ Maya เพื่อให้มีการอ่านข้อมูลของไฟล์ BVH เข้าไปประมวลผลสร้างเป็นรูปร่างของมนุษย์และแสดงการเคลื่อนไหว ซึ่ง MEL Script ที่ใช้นั้นมาจาก [W072]

- ส่วนหนึ่งที่เกี่ยวข้องกับขั้นตอนการกรองข้อมูลจากการเคลื่อนไหวที่แปลงข้อมูลจากองศาในการหมุน (Rotation) มาเป็นตำแหน่งที่เกิดขึ้นจริงในสภาพแวดล้อมนั้น (World Co-ordinate) ได้ถูกคำนวณไปพร้อมๆกับการอ่าน MEL Script เข้าไปใน Maya โดยที่เป็นการใช้ชุดคำสั่งที่อยู่ใน MEL Script นั้นแปลงให้เป็นตำแหน่งที่เกิดขึ้นจริงในสภาพแวดล้อมและเขียนออกไปในไฟล์ตัวอักษรเพื่อเตรียมใช้ต่อไปในระบบ โดยคำสั่งที่เรียกใช้คือ

  xform -ws -q -t $value

  และค่าพารามิเตอร์ที่แทนเข้าไปได้แก่

ws/-worldSpace	(C) treat values as world-space transformation values (only works for pivots, translations, rotation, rotation axis, matrix, and bounding box flags)
-q/-query	query transformation values
-t/-translation double double double	(C, Q) translation
$value	Rotation Value

4.1 ขั้นตอนการอ่านข้อมูลการเคลื่อนไหวของมนุษย์ (Read MoCap Data)

เมื่อมองเข้าไปในขั้นตอนการทำงานของระบบนั้นเริ่มต้นจากการที่อ่านข้อมูลจากไฟล์ตัวอักษรที่ผ่านขั้นตอนการเตรียมการแล้วเข้าไประบบโดยเก็บไว้ในออปเจค PositionValue ก่อนเพื่อความสะดวกในการเรียกใช้ การที่เราค้นหาจุดเชื่อมต่อระหว่างภาพและเสียง (Spotting Session) ในขั้นตอนนี้สิ่งที่จำเป็นคือความเข้าใจในรูปแบบของข้อมูล MoCap ที่เรานำมาใช้ โดยในวิทยานิพนธ์ชิ้นนี้ได้นำเอารูปแบบไฟล์ .BVH มาใช้โดยเนื้อหาที่อยู่ในไฟล์นั้นประกอบด้วย

1. ลำดับชั้นของข้อต่อที่ทำการเก็บข้อมูล (Joint Hierarchy)
2. ลำดับในการเก็บข้อมูลในแต่แกน (X,Y,Z Ordering)
3. ความห่างระหว่างแต่ละข้อต่อ (Offset)
4. ตำแหน่งของข้อต่อหลักที่มีการเคลื่อนที่ (Root Position)
5. องศาการหมุนของข้อต่อโดยสัมพันธ์กับข้อต่อในลำดับชั้นที่สูงกว่า (Joint Rotation referred to its parent)
6. จำนวนเฟรมทั้งหมดที่เก็บในไฟล์
7. ความยาวของแต่ละเฟรม หน่วยเป็นวินาที
8. ข้อมูลบันทึกตำแหน่งของ 4. และ 5. ที่มีการเปลี่ยนแปลงไปในแต่ละเฟรม

เมื่อเราทราบถึงรูปแบบการเก็บข้อมูลนั้นๆแล้วก็จึงทำการอ่านค่าเข้ามาใช้ในระบบ ในการหาตำแหน่งที่เกิดขึ้นจริงในสภาพแวดล้อมนั้นมิได้ทำการหาจากระบบ หากแต่ใช้ความสามารถของแอพพลิเคชั่น Maya เนื่องด้วย Maya นั้นสามารถอ่านรูปแบบไฟล์ .BVH ได้อยู่แล้วจึงเพียงแต่อ่านค่าจากไฟล์ .BVH ไปแล้วทำการแปลงข้อมูลเป็นไฟล์ตัวอักษรอีกไฟล์หนึ่งเท่านั้นเพื่อนำไปใช้ต่อในระบบที่สร้างขึ้นใหม่

เอาท์พุทจากระบบเมื่อจบขั้นตอนนี้

```
-4.541305|34.22879|-9.757853|-7.087917|-4.485015|-18.780489|31.423222|-
28.446146|8.861509|-27.363623|41.695915|18.841974|1.008454|-6.416192|6.519274|-
29.38328|4.675504|-2.631628|6.083052|39.809933|-2.911508|15.679157|7.370433|-
8.377286|4.723315|10.528112|5.041691|7.963229|-.851295|7.534328|15.51936|2.928838|-
3.924501|26.283697|-132.272003|13.318242|10.02226|-13.293984|-4.198166|3.199633|-
4.674591|10.403776|-13.32119|-19.330254|-6.843849|-71.2435|-117.082001|-61.872101|-
13.872063|-9.352996|6.825864|-3.589788|3.122182|14.168723|2.232313|-5.180964|-
3.391413-4.967764|33.849056|-10.542427|-6.469659|-4.05953|-21.141392|34.156723|-
29.151039|10.931151|-29.05666|42.145439|20.701843|0.053031|-5.901634|6.495006|-
28.872002|3.53334|-1.484977|6.381377|40.9884|-7.48864|22.104046|-2.202681|-
5.89902|3.594808|10.864945|4.588177|5.752679|-.457075|7.554091|10.610643|2.138525|-
5.661782|24.003483|-131.167831|9.289845|10.689674|-13.291468|-3.54425|4.87265|-
2.889739|11.454887|-11.247|-19.902922|-4.960466|-67.48494|-119.584663|-59.192955|-
13.257893|-9.447616|7.258677|-4.188767|3.177728|14.487819|2.619286|-6.111164|-
```

ตัวอย่างไฟล์ BVH

```
HIERARCHY
ROOT Hips
{
	OFFSET	0.000000	0.000000	0.000000
	CHANNELS 6 Xposition Yposition Zposition Zrotation Xrotation Yrotation
	JOINT LeftHip
	{
		OFFSET	4.740440	-3.195970	-0.322709
		CHANNELS 3 Zrotation Xrotation Yrotation
		JOINT LeftKnee
		{
			OFFSET	-1.605680	-15.592300	1.005510
			CHANNELS 3 Zrotation Xrotation Yrotation
			JOINT LeftAnkle
			{
				OFFSET	-0.632401	-16.421000	-1.413730
				CHANNELS 3 Zrotation Xrotation Yrotation
				End Site
				{
					OFFSET	0.008423	-3.021750	-1.660875
				}
			}
		}
	}
	JOINT RightHip
	{
		OFFSET	-4.740440	-3.195970	-0.322709
		CHANNELS 3 Zrotation Xrotation Yrotation
		JOINT RightKnee
		{
			OFFSET	1.605680	-15.592300	1.005510
			CHANNELS 3 Zrotation Xrotation Yrotation
			JOINT RightAnkle
			{
				OFFSET	0.632401	-16.421000	-1.413730
				CHANNELS 3 Zrotation Xrotation Yrotation
				End Site
				{
					OFFSET	-0.008423	-3.021750	-1.660875
				}
			}
		}
	}
	JOINT Chest
	{
		OFFSET	0.000000	3.468200	0.078401
		CHANNELS 3 Zrotation Xrotation Yrotation
		JOINT LeftCollar
		{
			OFFSET	3.857650	15.070700	-0.011583
			CHANNELS 3 Zrotation Xrotation Yrotation
			JOINT LeftShoulder
			{
				OFFSET	3.979720	-1.940530	-0.165223
				CHANNELS 3 Zrotation Xrotation Yrotation
				JOINT LeftElbow
				{
					OFFSET	0.615844	-10.986600	-0.673886
					CHANNELS 3 Zrotation Xrotation Yrotation
					JOINT LeftWrist
					{
						OFFSET	0.826843	-10.854500	1.034220
```

```
                                                CHANNELS 3 Zrotation Xrotation Yrotation
                                                End Site
                                                {
                                                        OFFSET   -1.679080 -8.456540 0.936217
                                                }
                                        }
                                }
                        }
                        JOINT RightCollar
                        {
                                OFFSET   -3.857650 15.070700 -0.011583
                                CHANNELS 3 Zrotation Xrotation Yrotation
                                JOINT RightShoulder
                                {
                                        OFFSET   -3.979720 -1.940530 -0.165223
                                        CHANNELS 3 Zrotation Xrotation Yrotation
                                        JOINT RightElbow
                                        {
                                                OFFSET   -0.615845 -10.986600            -0.673886
                                                CHANNELS 3 Zrotation Xrotation Yrotation
                                                JOINT RightWrist
                                                {
                                                        OFFSET   -0.826843 -10.854500            1.034220
                                                        CHANNELS 3 Zrotation Xrotation Yrotation
                                                        End Site
                                                        {
                                                                OFFSET   1.679080   -8.456540 0.936217
                                                        }
                                                }
                                        }
                                }
                        }
                        JOINT Neck
                        {
                                OFFSET   0.000000   15.101400 0.055730
                                CHANNELS 3 Zrotation Xrotation Yrotation
                                JOINT Head
                                {
                                        OFFSET   0.000000   4.185120   1.488260
                                        CHANNELS 3 Zrotation Xrotation Yrotation
                                        End Site
                                        {
                                                OFFSET   0.000000   7.773420   -0.106016
                                        }
                                }
                        }
                }
}
MOTION
Frames:    514
Frame Time: 0.016667
-0.084595 38.246128 -0.225864 0.041946   0.162033   0.493056   0.039115   -0.086300 -0.356226 -0.034828 -0.731111
0.021933   -0.046908 0.655254   -0.158153 0.319309   0.096342   -0.008434 -0.065851 -1.062675 -0.751527 -0.307313
0.800641   0.271745   -0.648075 -0.259187 0.945731   8.123368   -3.828663 -2.294878 61.059776 4.021795   16.448599
9.256327   -1.205854 22.978506 18.622736 0.739898   -38.269436 -7.103603 -1.562360 0.762083   -65.016464 10.902160 -
26.575441 -8.301103 3.776896   14.893243 -14.457111 -6.090618 20.527565 2.005794 3.765641   3.441903   -0.822275
5.377624   -4.537976

-0.084595 38.246128 -0.225864 0.041946   0.162033   0.493056   0.039115   -0.086300 -0.356226 -0.034828 -0.731111
0.021933   -0.046908 0.655254   -0.158153 0.319309   0.096342   -0.008434 -0.065851 -1.062675 -0.751527 -0.307313
0.800641   0.271745   -0.648075 -0.259187 0.945731   8.123368   -3.828663 -2.294878 61.059776 4.021795   16.448599
9.256327   -1.205854 22.978506 18.622736 0.739898   -38.269436 -7.103603 -1.562360 0.762083   -65.016464 10.902160 -
26.575441 -8.301103 3.776896   14.893243 -14.457111 -6.090618 20.527565 2.005794 3.765641   3.441903   -0.822275
5.377624   -4.537976
```

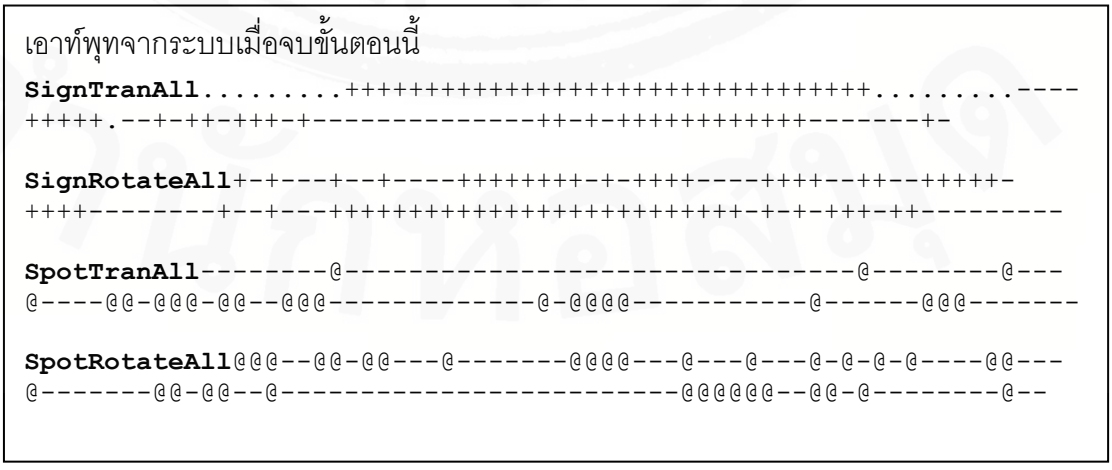
4.2 ขั้นตอนการหาจุดข้อต่อที่สำคัญ (Average All Joint Transition)

เราทำการวิเคราะห์ข้อมูลในรูปแบบของกราฟการเคลื่อนไหวโดยประกอบด้วยกราฟที่ได้จากข้อต่อแต่ละข้อต่อโดยมีแกน X แทนช่วงเวลาหรือลำดับเฟรมและแกน Y แทนตำแหน่งที่มีการเปลี่ยนแปลงไป เพื่อสะดวกต่อการนำไปใช้เราจึงได้ทำการรวมกราฟทั้งหมดนั้นเข้าไปกราฟเพียงเส้นเดียวโดยการรวมที่เกิดนั้นรวมจากค่าเฉลี่ยซึ่งไม่มีผลมากนักหากนำเอากราฟที่มีการเปลี่ยนแปลงน้อยมารวมกับกราฟที่มีการเปลี่ยนแปลงมาก เพราะกราฟที่ได้ยังเพียงในการถอดข้อมูลไปใช้หาจุดสนใจได้และยังสามารถสะท้อนภาพรวมของการเคลื่อนไหวนั้นๆได้ดีกว่าการใช้กราฟเพียงบางเส้น

4.3, 4.4 ขั้นตอนการหาจุดเปลี่ยนแปลงในการเคลื่อนไหวและจุดเชื่อมต่อกันเสียง (Detect Sign, Detect Spotting Session)

จากนั้นจึงทำการหาจุดสนใจโดยอัลกอริทึมที่ใช้ในการหานั้นออกแบบจากสมมุติฐานที่สนใจถึง “การเปลี่ยนแปลงในข้อต่อส่วนปลาย” โดยหากมองการเคลื่อนไหวหนึ่งๆที่เกิดขึ้นนั้นยกตัวอย่างเช่นการเดินหรือการต่อสู้ การที่เสียงเกิดขึ้นนั้นส่วนใหญ่เกิดขึ้นจากการเคลื่อนไหวที่ส่วนปลายของร่างกายได้ยืดออกไปจนสัมผัสกับสิ่งแวดล้อมต่างๆเช่น เท้าสัมผัสกับพื้นหรือการเตะต่อยโดยสิ่งของ หากเราพิจารณาจากข้อมูลที่ได้มานั้นเป็นเพียงข้อมูลของการเคลื่อนไหวเดี่ยวๆที่ไม่มีข้อมูลของสิ่งแวดล้อมมาเกี่ยวข้องแต่เราสามารถหาความหมายโดยนัยที่ว่าเมื่อส่วนปลายนั้นๆได้ทำการสัมผัสให้เกิดเสียงแล้วมีการเคลื่อนไหวกลับไปในทางตรงข้าม ซึ่งข้อมูลที่มีอยู่นี้เพียงพอต่อสมมุติฐานนี้ เมื่อใส่ข้อมูลเข้าไปในระบบแล้ว ระบบทำการค้นหาโดยเปรียบเทียบข้อมูลระหว่างแต่ละเฟรมว่ามีการเคลื่อนไหวเกิดขึ้นตามเงื่อนไขนี้หรือไม่ แล้วทำการเก็บไว้ในออปเจค SSValue เพื่อนำไปใช้ต่อไป

เอาท์พุทจากระบบเมื่อจบขั้นตอนนี้



4.5 ขั้นตอนการเชื่อมต่อระหว่างการเคลื่อนไหวและดนตรี

(Map MoCap and MuCom Parameter)

ในส่วนของการเชื่อมพารามิเตอร์ระหว่าง MoCap และ MuCom นั้น มีการสร้างออปเจค NoteValue เพื่อใช้ในการสร้างโน้ตดนตรี หากอ้างอิงจากลำดับชั้นของโมเดลในการสร้างดนตรีที่ได้ออกแบบไว้ข้อมูลนี้อยู่ในเลเวล 1 ซึ่งประกอบด้วย

1. Pitch หรือระดับความถี่สูงต่ำของเสียง
2. Timing หรือเวลาที่มีเสียงเกิดขึ้น
3. Velocity หรือน้ำหนักความดังเบาของเสียง
4. Duration หรือช่วงเวลาที่เสียงนั้นๆดังค้างอยู่

ตารางที่ 4.1

ระดับขั้นในการเพิ่มความซับซ้อนของดนตรี

ระดับ	*ส่วนที่เกี่ยวข้อง*
Level 3	Chord Progression
Level 2	Key Signature, Chord
Level 1	**Pitch, Duration, Timing, Velocity**

ตารางที่ 4.2

พารามิเตอร์ที่ถูกกำหนดให้เชื่อมโยงกัน

MuCom	*MoCap*	*คำอธิบาย*
Pitch	Amplitude	จากข้อมูลของ MoCapนั้นแอมพลิจูดที่เกิดขึ้นในแต่ละจุดที่เลือกมาเป็นจุดสนใจถูกเก็บไว้เพื่อใช้แมพเป็นระดับเสียงใน MuCom ต่อไป
Timing	KeyFrame	ในที่นี้หมายถึงหมายเลขของเฟรมที่เลือกให้เป็นจุดสนใจ โดยนำไปใช้กำหนดเวลาที่มีเสียงดนตรีเกิดขึ้น
Duration	n/a	ค่านี้ถูกคำนวณขึ้นภายหลังเพื่อให้เหมาะสมกับจำนวนโน้ตที่เกิดขึ้น

พารามิเตอร์ที่แสดงถึงระดับความถี่ของดนตรีนั้นมีความสำคัญที่สุดในการประพันธ์ดนตรี ในวิทยานิพนธ์นี้ได้ทำการพิจารณาถึงพารามิเตอร์ที่เหมาะสมในการนำเสนอซึ่งเชื่อมต่อให้มีความสัมพันธ์กับแอมปลิจูดของกราฟการเคลื่อนไหว โดยทั่วไปแล้วแอมปลิจูดของกราฟดนตรีจะส่งผลต่อความดังของเสียง โดยที่ความถี่ของกราฟจะส่งผลต่อระดับความถี่เสียง แต่ในวิทยานิพนธ์นี้ได้ใช้แอมปลิจูดของกราฟการเคลื่อนไหวมาแทนระดับความถี่เสียงเพื่อให้ตอบสนองกับการเคลื่อนไหวมากกว่า เนื่องจากในการเคลื่อนไหวของมนุษย์นั้นการเปลี่ยนแปลงแอมปลิจูดเกิดขึ้นสอดคล้องกับเสียงมากกว่าการเปลี่ยนแปลงทางความถี่ โดยที่ค่าของแอมปลิจูดของกราฟการเคลื่อนไหวนั้นจะแสดงตำแหน่งที่มีการเคลื่อนไหวสูงต่ำหรือไกล้ไกล กล่าวอีกนัยหนึ่งคือในการเคลื่อนที่สำหรับการเล่นเครื่องดนตรีนั้นก็เป็นไปในกรณีเดียวกันเช่นเปียโนหรือกีต้าร์จะมีโน้ตเสียงสูงอยู่ทางด้านขวามือ หรือเครื่องเป่าจะมีโน้ตเสียงสูงอยู่ทางด้านบน

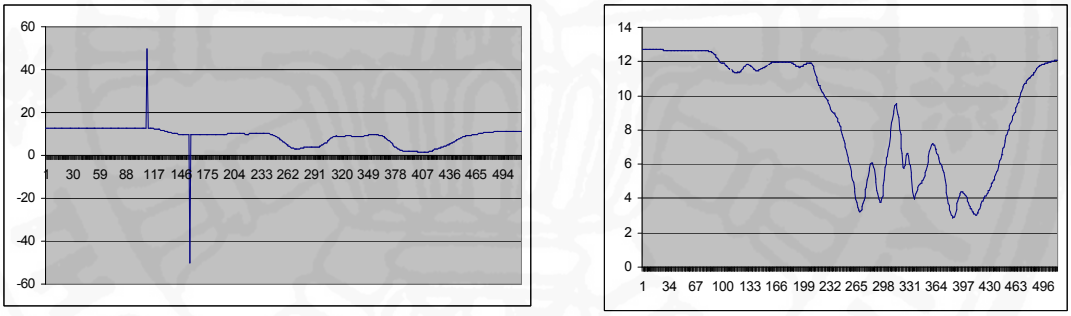
โดยพารามิเตอร์ที่กำหนดเพื่อเชื่อมความสัมพันธ์กันนั้นได้ระบุไว้ตามตารางที่ 4.2 ซึ่งในระดับของข้อมูลนี้สามารถสร้างเอาท์พุทออกไปเป็นไฟล์ MIDI โดยอาศัย Method จาก Library ของ JMUSIC ได้ทันที แต่หากว่าเราต้องการได้ดนตรีในระดับที่ซับซ้อนขึ้นจำเป็นต้องพัฒนาต่อในระดับที่สูงขึ้นโดยยึดตามข้อมูลที่ได้จากระดับล่าง

4.6 ขั้นตอนการกรองข้อมูลที่ไม่ต้องการออกไป (Compress Data)

ในส่วนของการกรองข้อมูลที่ไม่ต้องการออกไปนั้นได้เสริมไว้ในกรณีที่หากข้อมูลที่มีอยู่นั้นมีส่วนที่ไม่เหมาะสมกับอัลกอริทึมที่ใช้ในการประมวลผล โดยในการทำงานของระบบนั้นเหมาะกับข้อมูลที่มีการกระจายในช่วงที่ไกล้ๆกันมากกว่าข้อมูลที่มีค่าโดดเด่นเกินไป สังเกตุจากรูปได้ว่าข้อมูลที่ Peak เกินไปเมื่อเทียบกับสมาชิกตัวอื่นนั้นถูกคัดออกไปจากระบบก่อน เพื่อให้สมาชิกตัวที่เหลือได้กระจายไปอยู่ทั่วขอบเขต ผลกระทบที่เกิดจากการที่ไม่ทำการกรองข้อมูลนั้น ยกตัวอย่างจากรูปทางซ้ายด้วยสมาชิกบางตัวที่มีค่าสูงหรือต่ำเกินไปนั้นทำให้สมาชิกตัวที่เหลือกระจุกอยู่ในตำแหน่งเกือบเป็นค่าเดียวกัน ผลลัพท์คือโน้ตดนตรีเพียง 3-4 ตัวเท่านั้นซึ่งรูปทางขวานั้นมีการกระจายอย่างเหมาะสมทำให้มีการสร้างโน้ตได้จำนวนเสียงที่มากขึ้น

ภาพที่ 4.2

กราฟการเคลื่อนไหวก่อนและหลังการกรองข้อมูล



4.7 ขั้นตอนการแปลงช่วงขอบเขต (Convert Interval)

จากนั้นคือการกำหนดขอบเขตของเสียงที่เป็นไปได้ (Range) ในวิทยานิพนธ์นี้ถือเป็นเรื่องสำคัญเมื่อเทียบกับงานวิจัยอื่นๆที่ไม่ได้กำหนดถึงเรื่องนี้ เครื่องดนตรีอย่างเช่นเปียโนสามารถสร้างเสียงความถี่ได้แตกต่างกันถึง 88 เสียงแต่โดยธรรมชาติของมนุษย์นั้นสามารถสร้างได้จำกัดกว่านั้นมาก จากตารางที่ 4.3 เป็นการกำหนดขอบเขตของเสียงของนักร้องประสานเสียงแต่ละตำแหน่งซึ่งใช้เพียง 22 เสียงเท่านั้น ในวิทยานิพนธ์นี้กำหนดขอบเขตโดยใช้เสียงของนักร้องชาย (Male Voice) ซึ่งใช้ 29 เสียงโดยเริ่มจาก E2 – A4 ความหมายของตัวเลขนี้ถูกอธิบายไว้ในย่อหน้าถัดไป

ตารางที่ 4.3

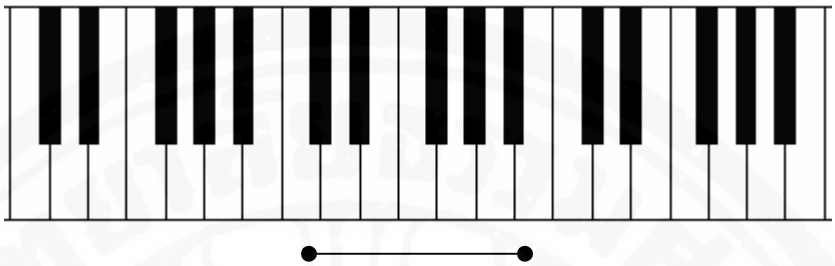
ช่วงห่างของเสียงร้องในทางดนตรี

เพศของผู้ร้อง	*ตำแหน่งของผู้ร้อง*	*ขอบเขตของเสียง*
Male Voice	Bass	E2 – D4
	Tenor	B2 – A4
Female Voice	Alto	G3 – F5
	Soprano	D4 – C6

โดยปกติแล้วโน้ตดนตรีมีเพียง 12 ตัวเท่านั้นโดยในแต่ละชุดของเสียงโน้ต 12 ตัวที่เสียงสูงต่ำต่างกันนั้นเรียกเป็นออคเตฟ (Octave) โน้ตตัวเดียวกันแต่อยู่คนละออคเตฟมีความถี่ที่กลมกลืนกัน (Harmonic) อ้างอิงจากภาพของคีย์บนเปียโนนี้มี 3 ออคเตฟซึ่งในแต่ละออคเตฟมีอยู่ 12 ตัวโน้ตโดยเริ่มจากโน้ตตัว C ไปจนถึงตัว B โดยเลขที่ต่อกันนั้นหมายถึงตัวเลขของออคเตฟ

ภาพที่ 4.3

แสดงออคเตฟและคีย์ที่ใช้สร้างโน้ตแต่ละตัว



ในตารางที่ 4.4 เพิ่มเติมข้อมูลในส่วนที่สอดคล้องกันกับออคเตฟที่เพิ่งกล่าวถึงไป ในเรื่องของค่าสำหรับตัวโน้ตนั้นสามารถมองได้ในรูปแบบต่างๆกันไปตามวัตถุประสงค์ที่ใช้แต่ขอให้เทียบจากตารางนี้ได้ว่าคือค่าเดียวกัน

- เริ่มจากที่บรรทัดแรกในตารางนั้นตรงกับโน้ตทั้ง 12 ที่มีอยู่ใน 1 ออคเตฟ (เทียบกับรูปคีย์ของเปียโนได้เพื่อสร้างความคุ้นเคยมากขึ้น) โดยที่โน้ตที่ติดกันนั้นมีความห่างของเสียงเท่ากันหมดคือ ครึ่งเสียง (Semi-tone)
- บรรทัดถัดมาเทียบโน้ตทั้ง 12 ตัวนั้นเข้ากับชื่อ โดยที่ชื่อนี้ถูกนำไปใช้อ้างอิงต่อไปกับคีย์หลัก
- เมื่อเพิ่มเนื้อหาเกี่ยวกับสเกล (Scale) เข้าไปด้วย หมายเลขในที่นี้หมายถึงโน้ตตัวหลักๆที่เล่น ในหนึ่งคีย์หลัก (เทียบกับรูปคีย์ของเปียโน หากเล่นในคีย์ C สามารถเล่นเป็นคีย์สีขาวเท่านั้นโดยที่ไม่เล่นคีย์สีดำเลย) ความแตกต่างระหว่างดนตรีไทยและดนตรีสากลคือความห่างของเสียงแต่ละเสียงโดยที่ดนตรีไทยห่างเท่าๆกันหมดแต่ดนตรีสากลมีทั้งเต็มเสียงและครึ่งเสียงตามสเกลที่ใช้ สังเกตได้ที่โน้ตตัวที่ 3 และ 4 , 7 และ 1 (ติดกับโน้ตตัวที่ของออคเตฟถัดไป)
- ในตารางส่วนที่เหลือนั้นหมายถึงค่าที่ใช้ในระบบ เพื่ออ้างอิงว่าเป็นโน้ตตัวใดและอยู่ในออคเตฟไหน จากที่เราใช้ขอบเขตของเสียงที่อยู่ในช่วง E2 – A4 นั้นในระบบกำหนดเป็น 28 – 57 ไว้ รวมถึงโน้ตที่ประมวลผลอยู่ในรูปแบบนี้เช่นเดียวกัน

ตารางที่ 4.4

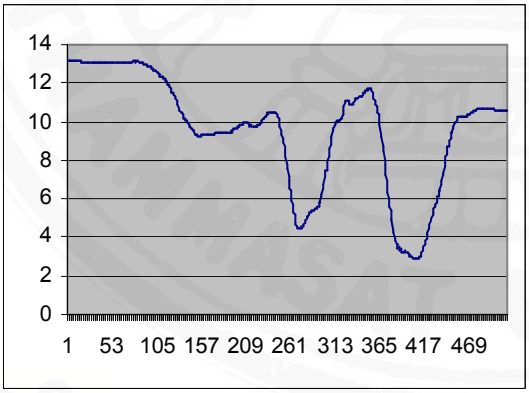
ค่าของตัวโน้ตที่แสดงในเนื้อหาที่ต่างกัน

โน้ตตัวที่	***1***	***2***	***3***	***4***	***5***	***6***	***7***	***8***	***9***	***10***	***11***	***12***
ชื่อของโน้ต	***C***	***C#/Db***	***D***	***D#/Eb***	***E***	***F***	***F#/Gb***	***G***	***G#/Ab***	***A***	***A#/Bb***	***B***
โน้ตที่เล่นในสเกล	***1***		***2***		***3***	***4***		***5***		***6***		***7***
Octave 0	0	1	2	3	4	5	6	7	8	9	10	11
Octave 1	12	13	14	15	16	17	18	19	20	21	22	23
Octave 2	24	25	26	27	**28**	29	30	31	32	33	34	35
Octave 3	36	37	38	39	40	41	42	43	44	45	46	47
Octave 4	48	49	50	51	52	53	54	55	56	**57**	58	59
Octave 5	60	61	62	63	64	65	66	67	68	69	70	71
Octave 6	72	73	74	75	76	77	78	79	80	81	82	83
Octave 7	84	85	86	87	88	89	90	91	92	93	94	95
Octave 8	96	97	98	99	100	101	102	103	104	105	106	107
Octave 9	108	109	110	111	112	113	114	115	116	117	118	119
Octave 10	120	121	122	123	124	125	126	127				

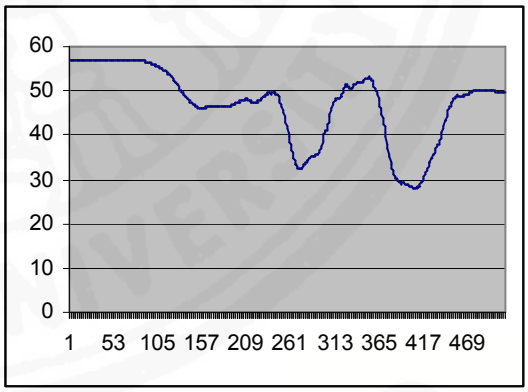
ภาพที่ 4.4

กราฟการเคลื่อนไหวที่แปลงไปเป็นกราฟดนตรี

MoCap Graph



MuCom Graph



เมื่อกำหนดขอบเขตของเสียงได้แล้วนั้น ระบบนำเอากราฟการเคลื่อนไหวที่ได้นั้นมาแมพลงไป โดยที่จุดที่มีค่าสูงสุงเท่ากับโน้ตที่เสียงสูงที่สุด และในทำนองเดียวกันกับเสียงต่ำ โดยเมื่อเราทราบค่าของตำแหน่งของ v1 โดยที่

$$v1 \in \{\min 1, ..., \max 1\}$$

v1 = ค่าของแอมปลิจูด ณ จุดสนใจซึ่งเป็นสมาชิกของจำนวนที่อยู่ภายในขอบเขตของ MoCap Interval โดยมีค่าระหว่าง min1 และ max1

min1 = ค่าที่น้อยที่สุดที่อยู่ใน MoCap Interval

max1 = ค่าที่มากที่สุดที่อยู่ใน MoCap Interval

เมื่อต้องการหาตำแหน่งของ v2 โดยที่

$$v2 \in \{\min 2, ..., \max 2\}$$

v2 = ค่าของระดับเสียงของโน้ต ณ จุดสนใจซึ่งเป็นสมาชิกของจำนวนที่อยู่ภายในขอบเขตของ MuCom Interval โดยมีค่าระหว่าง min2 และ max2

min2 = ค่าที่น้อยที่สุดที่อยู่ใน MuCom Interval

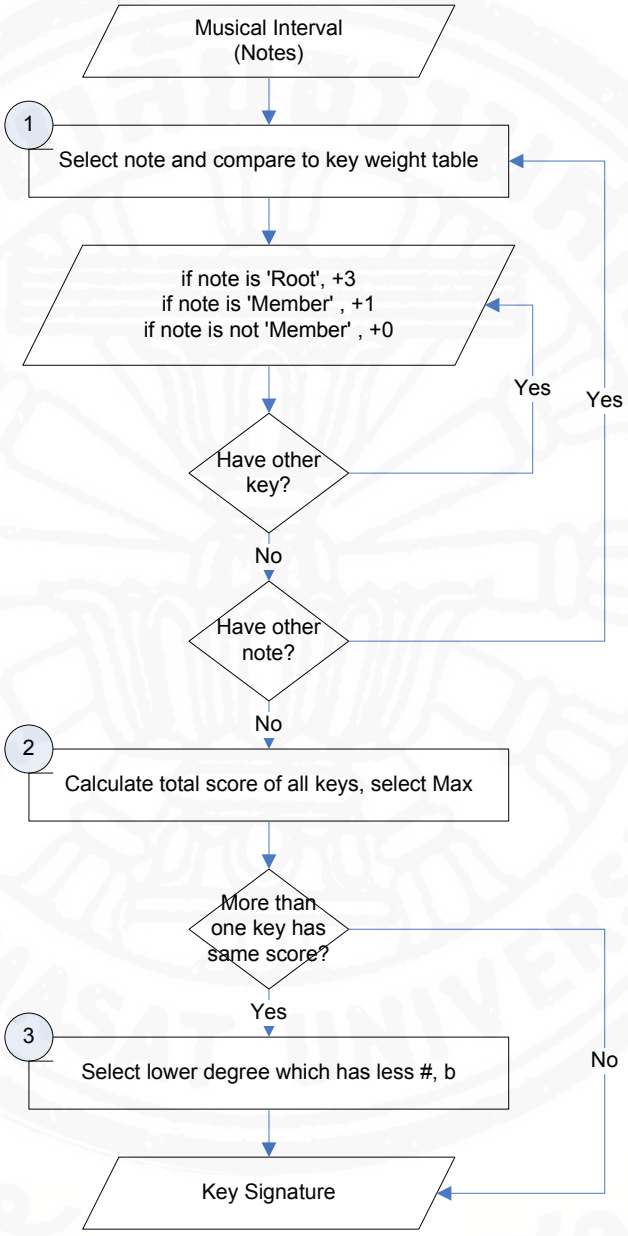
max2 = ค่าที่มากที่สุดที่อยู่ใน MuCom Interval

สามารถคำนวณได้โดยกำหนดค่าเข้าไปในสูตรต่อไปนี้เพื่อทำการเปลี่ยนให้กราฟการเคลื่อนไหวอยู่ในรูปแบบของกราฟดนตรี

$$v2 = \left[\frac{(v1 - \min 1)}{(\max 1 - \min 1)} \times (\max 2 - \min 2) \right] + \min 2$$

ผลลัพท์ที่ได้ในจุดนี้คือโน้ตส่วนที่ใช้เล่นทำนองของมือขวา (Right-Hand Note) โดยที่โน้ตทุกตัวที่ได้นั้นถูกนำมาใช้ประมวลผลในขั้นตอนถัดไปคือการหาคีย์หลักซึ่งมีรายละเอียดการทำงานดังภาพ 4.5

ภาพที่ 4.5
รายละเอียดของขั้นตอนในการหาคีย์หลัก



เอาท์พุทจากระบบก่อน Convert Interval

12.645223|12.605811|12.605298|12.608231|12.606591|12.606305|12.606
362|12.606367|12.605785|12.604895|12.603727|12.599458|12.598007|12
.603839|12.687952|12.686929|12.679360|12.687429|12.681101|12.51864

เอาท์พุทจากระบบหลัง Convert Interval

56|56|56|56|56|56|56|56|56|56|56|56|56|56|57|56|56|56|56|56|56|56|
56|49|49|48|49|49|49|49|49|49|50|50|49|50|50|50|50|50|50|31|34|34|
46|47|47|47|47|47|47|47|47|47|49|30|29|29|29|29|29|29|29|29|29|28|

4.8 ขั้นตอนในการหาคีย์หลัก (Find Key Signature)

อ้างอิงจากตารางที่ 4.5 นั้นเป็นการเพิ่มระดับในการประพันธ์ดนตรีไปอีกหนึ่งขั้นโดยใช้ข้อมูลที่เป็นออปเจค NoteValue ที่ได้มาจากเลเวลที่ 1 มาทำการหาคีย์หลักและคอร์ด

ตารางที่ 4.5

ระดับขั้นในการเพิ่มความซับซ้อนของดนตรี

ระดับ	*ส่วนที่เกี่ยวข้อง*
Level 3	Chord Progression
Level 2	**Key Signature, Chord**
Level 1	Pitch, Duration, Timing, Velocity

1. Key Signature คือสัญลักษณ์ที่อยู่ในส่วนเริ่มต้นของโน้ตดนตรีเพื่อการกำหนดโน้ตตัวเริ่มต้นและโน้ตที่สามารถใช้ได้ [H01] โดยมีการบอกถึงชาร์ป (#) และแฟลต (b) ที่บังคับใช้อยู่ในเพลง เครื่องหมายทั้งสองนี้เรียกว่าแอคซิเดนทอล (Accidental) สำหรับโน้ตที่อยู่ในบรรทัดที่มีเครื่องหมายแอคซิเดนทอลนี้กำกับอยู่ถูกบังคับให้เพิ่มหรือลดไปครึ่งเสียง ประโยชน์ของคีย์หลักนั้นยังช่วยลดการเขียนแอคซิเดนทอลหลายๆตัวลงไปในโน้ต สังเกตุจากภาพที่ 4.6 นั้น มีการเขียนเครื่องหมายแอคซิเดนทอลบังคับโน้ตหลายตัวมากซึ่งคีย์หลักสามารถใช้แทนได้ดังภาพที่ 4.7 โดยไม่จำเป็นต้องเขียน แอคซิเดนทอในทุกๆตัวโน้ตอีกต่อไป

ภาพที่ 4.6

โน้ตที่ไม่มีการกำหนดคีย์หลัก

ภาพที่ 4.7

สัญลักษณ์สำหรับคีย์ F#

F#

1. กระบวนการหาคีย์หลักนั้นเริ่มต้นจากการที่นำโน้ตทุกตัวที่ได้มาจากขั้นตอนที่แล้วมาทำการวิเคราะห์ว่าควรจัดให้อยู่ในคีย์หลักใด
    a. ระบบนับโน้ตทุกตัวโดยโน้ตเดียวกันที่อยู่ต่างออคเตฟก็นับว่าเป็นโน้ตตัวเดียวกันไปด้วย
    b. ระบบทำการคำนวณจำนวนตัวโน้ตกับน้ำหนักของแต่ละคีย์หลักที่ได้มาจากตารางแล้วบันทึกคะแนนไว้ ในแต่ละคีย์หลักนั้นเริ่มต้นด้วยโน้ตตัวแรกที่ต่างกันไป ดังนั้นโน้ตตัวเดียวกันที่อยู่ต่างคีย์หลักนั้นจึงมีน้ำหนักไม่เท่ากัน
    c. การให้คะแนนมีอยู่ 3 เงื่อนไขด้วยกันคือ 3,1 และ 0 ซึ่งเกณฑ์ในการกำหนดค่าที่ให้น้ำหนักนั้นเพื่อที่จะทราบความแตกต่างของน้ำหนักรวมที่ถูกเทียบกันในแต่ละคีย์เท่านั้นซึ่งไม่จำเป็นต้องใช้ค่าที่มากจนเกินไป
        i. ได้ 3 คะแนนเมื่อมีโน้ตเกิดขึ้นในตำแหน่งที่เป็นรูทของคีย์หลักนั้นๆจะได้คะแนนพิเศษกว่าเพื่อสร้างความสำคัญให้โดดเด่นกว่าโน้ตตัวอื่นๆ
        ii. ได้ 1 คะแนนเมื่อมีโน้ตเกิดขึ้นในตำแหน่งที่เป็นสมาชิกของคีย์หลักนั้นๆโดยเท่ากับนับหนึ่งตัวโน้ต
        iii. ได้ 0 คะแนนเมื่อมีโน้ตเกิดขึ้นในตำแหน่งที่ไม่ได้เป็นสมาชิกของคีย์หลักนั้นๆโดยจะไม่ได้คะแนน
2. เมื่อรวมคะแนน ให้เลือกคีย์หลักที่ได้คะแนนมากที่สุด ซึ่งหมายความว่าเป็นคีย์หลักที่สอดคล้องกับโน้ตที่ได้มากที่สุด
3. แต่ถ้าหากว่ามีคีย์หลักที่ได้คะแนนเท่ากันเกิดขึ้น ให้เลือกคีย์ที่มีจำนวนเครื่องหมายชาร์ปหรือแฟลต (Degree) น้อยกว่าเพื่อให้นักดนตรีอ่านได้ง่ายกว่า

ตารางที่ 4.6

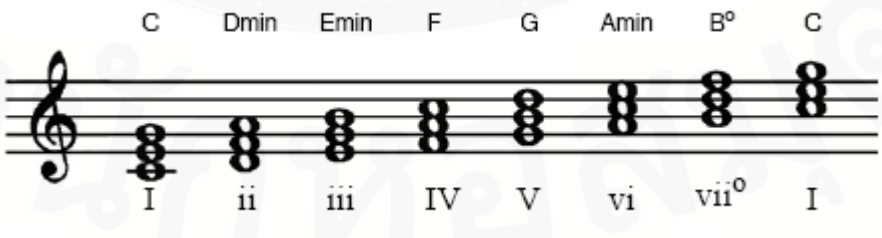
การให้น้ำหนักสำหรับตัวโน้ตในแต่ละคีย์

โน้ตตัวที่		*1*	*2*	*3*	*4*	*5*	*6*	*7*	*8*	*9*	*10*	*11*	*12*
ชื่อของโน้ต		*C*	*C#/Db*	*D*	*D#/Eb*	*E*	*F*	*F#/Gb*	*G*	*G#/Ab*	*A*	*A#/Bb*	*B*
โน้ตที่เล่นในสเกล		*1*		*2*		*3*	*4*		*5*		*6*		*7*
Key	***Degree***												
Ab	4	1	1	0	1	0	1	0	1	3	0	1	0
Eb	3	1	0	1	3	0	1	0	1	1	0	1	0
Bb	2	1	0	1	1	0	1	0	1	0	1	3	0
F	1	1	0	1	0	1	3	0	1	0	1	1	0
C	**0**	**3**	**0**	**1**	**0**	**1**	**1**	**0**	**1**	**0**	**1**	**0**	**1**
G	1	1	0	1	0	1	0	1	3	0	1	0	1
D	2	0	1	3	0	1	0	1	1	0	1	0	1
A	3	0	1	1	0	1	0	1	0	1	3	0	1
E	4	0	1	0	1	3	0	1	0	1	1	0	1
B	5	0	1	0	1	1	0	1	0	1	0	1	3
F# / Gb	6	0	1	0	1	0	1	3	0	1	0	1	1
C# / Db	7	1	3	0	1	0	1	1	0	1	0	1	0

2. Chord คือตัวโน้ตตั้งแต่สามตัวขึ้นไปที่มีเสียงเกิดขึ้นมาได้ยินพร้อมกัน รูปแบบของคอร์ดนั้นมีอยู่หลายแบบเพื่อใช้แทนตัวสมาชิกที่ใช้ในคอร์ดนั้นๆ การระบุคอร์ดนั้นสามารถใช้สัญลักษณ์ด้วยตัวหนังสือหรือตัวเลขโรมันก็ได้ ระบบต้องการทราบถึงรูทของแต่ละคอร์ดเพื่อกำหนดสมาชิกในคอร์ดนั้นๆ

ภาพที่ 4.8

คอร์ดในสเกล C Major



4.9 ขั้นตอนในการหาทางเดินคอร์ด (Chord Progression)

รูปแบบของคอร์ดที่ได้กำหนดไว้ในเลเวลที่ 2 นั้นถูกเตรียมไว้เพื่อบอกถึงสมาชิกที่ต้องการในการสร้างคอร์ดหากแต่ว่าสิ่งที่กำหนดว่ารูทที่ต้องการนั้นคือโน้ตตัวใดถูกกำหนดในเลเวลที่ 3 ในการหาทางเดินคอร์ดซึ่งหมายถึงแนวทางของคอร์ดที่ถูกเปลี่ยนแปลงไปโดยเชื่อมโยงกับคอร์ดที่แล้ว

ตารางที่ 4.7

ระดับขั้นในการเพิ่มความซับซ้อนของดนตรี

ระดับ	*ส่วนที่เกี่ยวข้อง*
Level 3	Chord Progression
Level 2	Key Signature, Chord
Level 1	Pitch, Duration, Timing, Velocity

ยกตัวอย่างเช่นเพลงๆหนึ่งถูกแบ่งเป็นห้องๆ (Bar) อาจเป็น 4, 8 หรือ 16 ห้อง การเปลี่ยนแปลงของคอร์ดในแต่ละห้องนั้นอาจเกิดขึ้นหลายครั้งในห้องเดียว หรืออาจไม่เปลี่ยนแปลงเลยจนเมื่อผ่านไปหลายๆห้อง รูปแบบของการประพันธ์ดนตรีนั้นอาจถูกสร้างโดยกำหนดทางเดินคอร์ดมาก่อน เช่นเมื่อเรากำหนดให้มี 8 ห้องแล้วจึงกำหนดคอร์ดที่ต้องการดังตารางที่ 4.9 โดยสามารถเปลี่ยนเป็นคีย์อะไรก็ได้เพียงกำหนดให้ I เป็นคีย์ที่ต้องการ

ตารางที่ 4.8

ตัวอย่างทางเดินคอร์ด

Bar	*1*	*2*	*3*	*4*	*5*	*6*	*7*	*8*
Progress	I	IV	V	I	V	IV	I	I

ในการหาทางเดินคอร์ดนั้น ระบบใช้การสุ่มคอร์ดต่อไปที่เหมาะสมให้จากความเป็นไปได้ของทางเดินคอร์ดที่ถูกใช้ทั่วไปโดยเริ่มต้นจากรูทเสมอ สัญลักษณ์ที่ใช้แทนทางเดินคอร์ดนั้นเป็นตัวเลขโรมันซึ่งหมายถึงสมาชิกแต่ละตัวในคีย์หลักนั้น ยกตัวอย่างถ้าเป็นคีย์ C คอร์ด V ของ C ก็คือ คอร์ด G เป็นต้น ในภาพที่ 4.9 แสดงว่าจากคอร์ดหนึ่งสามารถไปคอร์ดใดได้บ้างเช่น คอร์ด I สามารถไปได้ทุกคอร์ดยกเว้น vii เป็นต้น

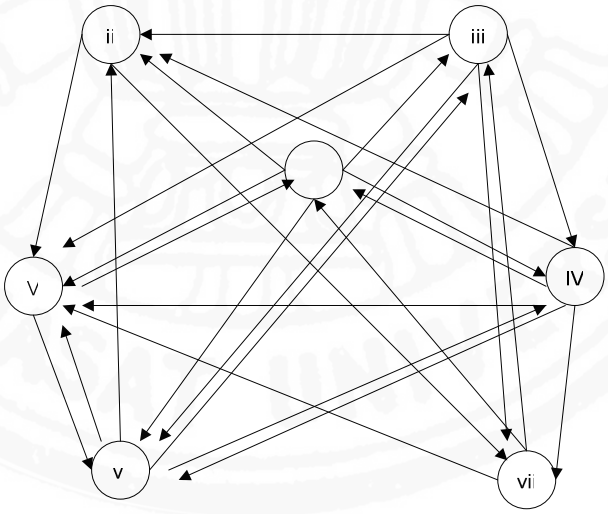
ตารางที่ 4.9

ค่าของตัวโน้ตในคีย์ C และเลขโรมันที่ใช้กับทางเดินคอร์ด

โน้ตตัวที่	***1***	***2***	***3***	***4***	***5***	***6***	***7***	***8***	***9***	***10***	***11***	***12***
ชื่อของโน้ต	***C***	***C#/Db***	***D***	***D#/Eb***	***E***	***F***	***F#/Gb***	***G***	***G#/Ab***	***A***	***A#/Bb***	***B***
โน้ตที่เล่นในสเกล	***1***		***2***		***3***	***4***		***5***		***6***		***7***
Key C	**I**		**ii**		**iii**	**IV**		**V**		**vi**		**vii**

ภาพที่ 4.9

ทางเดินคอร์ดที่เหมาะสม



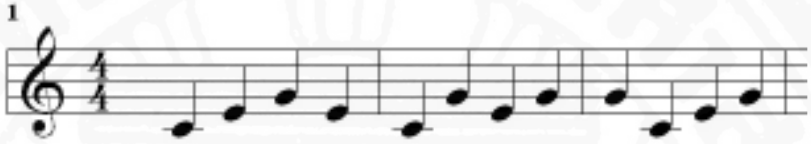
เอาท์พุทจากระบบเมื่อจบขั้นตอนนี้

```
Chord Progression = | I | V | vi | IV | vii | V | I | IV | V | vi
| V | I | V | I | vi | iii |
Chord Progression = | G# | D# | F | C# | G | D# | G# | C# | D# | F
| D# | G# | D# | G# | F | C |
```

4.10 ขั้นตอนในการหาลักษณะการเล่น (Chord Pattern)

หมายถึงลักษณะการเล่นสมาชิกแต่ละตัวของคอร์ด ยกตัวอย่างเช่นในการเล่นคอร์ด C ที่มีสมาชิก 3 ตัวนั้น อาจเล่นได้หลายรูปแบบดังภาพที่ 4.10

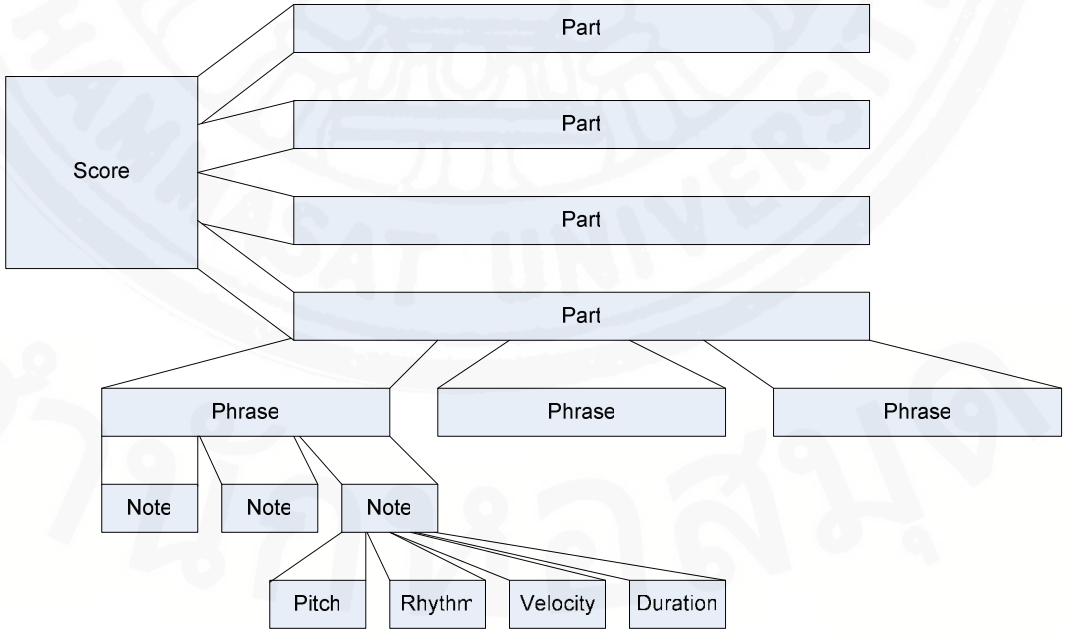
ภาพที่ 4.10

วิธีการเล่นในรูปแบบต่างๆ

4.11 ขั้นตอนในการสร้างโน้ตสำหรับเล่นด้วยมือขวา

ในการเรียกใช้ Library ของ JMUSIC นั้น มีการสร้างออปเจค Score หรือโน้ตของทั้งเพลงนั้นจากดนตรีส่วนย่อยๆหลายส่วนประกอบกันดังภาพที่ 4.11

ภาพที่ 4.11

ออปเจคที่ใช้ในการประกอบ Score



ออปเจคของ Score นั้นถูกสร้างด้วยออปเจคย่อยๆดังต่อไปนี้

1. Part คือ ส่วน หรือมองให้เข้าใจง่ายกว่านั้นคือส่วนที่เล่นโดยเครื่องดนตรีหนึ่งชิ้น เราสามารถแยกข้อมูลในส่วนนี้ไว้เพื่อกำหนด MIDI Channel ที่เป็นตัวเล่น Part นี้ รวมทั้งการควบคุมน้ำหนักดังเบาของเสียงด้วย เพราะโดยธรรมชาติแล้วเครื่องดนตรีแต่ละชิ้นให้ความถี่และความดังที่ต่างกันหากเราต้องการควบคุมความกลมกลืนของเครื่องดนตรีแต่ละชิ้นเราต้องกำหนดที่จุดนี้ ในวิทยานิพนธ์นี้ได้ทำการแยก Part ออกมา 3 ส่วน โดยส่วนแรกคือโน้ตที่เล่นโดยมือขวาที่ใช้บ่งบอกทำนองของดนตรี และอีกสองส่วนนั้นคือโน้ตที่เล่นด้วยมือซ้ายที่ใช้บ่งบอกคอร์ดและจังหวะของดนตรี

2. Phrase คือ วลี โดยดนตรีนั้นประกอบขึ้นจากวลีมากมาย บางวลีอาจเล่นซ้ำไปซ้ำมาหรืออาจเล่นตามแบบแผนของดนตรีแต่ละแนว สิ่งที่ได้รับจากการแปลง MoCap ไปเป็น MuCom นั้นคือวลีหนึ่งซึ่งสามารถปรับปรุงเปลี่ยนแปลง ลดเพิ่มตัวโน้ต หรือกระทั่งตัดต่อสลับสับเปลี่ยน เพื่อสร้างวลีที่พอใจขึ้นมาได้ ในวิทยานิพนธ์นี้นำเอาวลีนั้นมาวิเคราะห์เพื่อหาดนตรีส่วนอื่นมาเสริมให้มีความสมบูรณ์ตามแนวทางการแต่งเพลงทั่วไป

3. Note คือตัวโน้ต เป็นรายละเอียดในส่วนที่เล็กที่สุดและมีความสำคัญที่สุด Pitch, Rhythm, Velocity และ Duration ที่ประกอบขึ้นมาเป็นโน้ตนั้นสามารถถ่ายทอดอารมณ์ได้อย่างมีประสิทธิภาพ ในส่วนของ Pitch และ Rhythm นั้นค่อนข้างเป็นค่าที่เปลี่ยนแปลงน้อยซึ่งระบบสามารถรองรับได้ หากแต่ Velocity และ Duration นั้นเป็นเรื่องของอารมณ์และลักษณะนิสัยเฉพาะตัวของผู้เล่นจริงๆ

การที่แบ่งการประพันธ์โน้ตเป็นส่วนของมือขวาและมือซ้ายนั้นเพื่อที่ว่าโน้ตที่ได้สามารถเล่นได้ด้วยผู้เล่นเปียโนเพียงคนเดียว แต่โดยลักษณะของเอาท์พุทนั้นยังสามารถที่นำโน้ตในแต่ละส่วนนั้นแยกเป็นเครื่องดนตรีแต่ละชิ้นได้

โน้ตที่ใช้ในมือขวานั้นเป็นวลีที่ได้โดยตรงมาตั้งแต่ขั้นตอนที่ 7 (Convert Interval) ซึ่งเก็บอยู่ในออปเจค NoteValue สามารถสร้างได้โดยสร้างออปเจค Note เพื่อรับ NoteValue เข้าไปแล้วจึงใส่ไว้ใน Phrase, Part และ Score ตามลำดับ

4.12 ขั้นตอนในการสร้างโน้ตสำหรับเล่นด้วยมือซ้าย

ในส่วนของโน้ตสำหรับมือซ้ายนั้นมีอยู่อย่างน้อยสองส่วน คือส่วนที่บอกรูทของคอร์ดและอีกส่วนคือส่วนที่ใช้เล่นสนับสนุนมือขวา (Accompany) ความหมายในที่นี่ยังหมายความว่าควรมีเครื่องดนตรีอย่างน้อย 3 ชิ้นที่เล่นพร้อมๆกัน โดยชิ้นที่ 1 บอกทำนองเช่นแซคโซโฟนหรือกีต้าร์ที่มีเสียงโดดเด่นในการเล่นนำ ชิ้นที่ 2 เล่นคอร์ดเพื่อสนับสนุนและชิ้นที่ 3 เล่นเบสเพื่อคุมจังหวะของเพลง (ซึ่งในวิทยานิพนธ์นี้ได้ตัดบทบาทของกลองไป เบสจึงมีส่วนในการให้จังหวะ) โน้ตสำหรับมือซ้ายทั้งสองส่วนนั้นควรแยกเป็น 2 Parts เพื่อนำมาแยกเป็นเครื่องดนตรีต่างชนิดกันได้ หลังจากที่เราทราบถึงคีย์หลักแล้วเราก็สามารถสร้างโน้ตในส่วนมือซ้ายได้หลายวิธี ยกตัวอย่างเช่นในตารางที่ 4.10

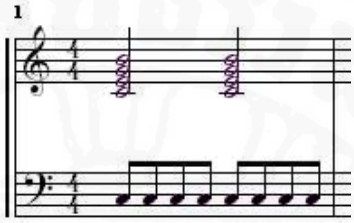
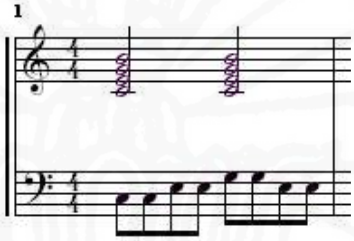
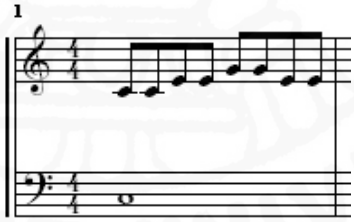
วิธีการสร้างโน้ตในที่นี้หมายถึงวิธีการเล่นที่จำเป็นต้องมีความลงตัวระหว่างความโดดเด่นและความเรียบง่ายให้ประสานสอดคล้องกัน ควรมีอยู่หนึ่ง Part ที่เล่นแบบง่ายที่สุดเพื่อเป็นรากฐานให้กับ Part อื่น

เอาท์พุทจากระบบเมื่อจบขั้นตอนนี้

```
In phrase-------- jMusic PHRASE: 'Untitled Phrase' contains 3
notes.  Start time: 0.0 --------
jMusic NOTE: [Pitch = 37][RhythmValue = 0.4375][Dynamic = 85][Pan
= 0.5][Duration = 0.4375]
jMusic NOTE: [Pitch = 41][RhythmValue = 0.4375][Dynamic = 85][Pan
= 0.5][Duration = 0.4475]
jMusic NOTE: [Pitch = 44][RhythmValue = 0.4375][Dynamic = 85][Pan
= 0.5][Duration = 0.4575]
```

ตารางที่ 4.10

วิธีในการสร้างโน้ตสำหรับมือซ้าย

Left-Hand Note Part 1	*Left-Hand Note Part 2*
เล่นคอร์ดค้างไว้	เล่นเบสเป็นจังหวะ
เล่นคอร์ดค้างไว้	เล่นเบสเป็นรูปแบบ
เล่นคอร์ดเป็นรูปแบบ	เล่นเบสค้างไว้

เอาท์พุทจากระบบเมื่อจบขั้นตอนนี้

```
------------------------------ Writing MIDI File ------------------
------------
Converting to SMF data structure...
    Part 0 'Untitled Part' to SMF Track on Ch. 0:  Phrase
0:.....................................................................
..................
    Part 1 'Untitled Part' to SMF Track on Ch. 0:  Phrase
0:.............................................
    Part 2 'Untitled Part' to SMF Track on Ch. 0:  Phrase 0:...
Phrase 1:... Phrase 2:... Phrase 3:... Phrase 4:... Phrase 5:...
Phrase 6:... Phrase 7:... Phrase 8:... Phrase 9:... Phrase 10:...
Phrase 11:... Phrase 12:... Phrase 13:... Phrase 14:... Phrase
15:...
MIDI file 'C:/Documents and Settings/Boing/Desktop/mucom.mid'
written from score 'Untitled Score' in 0.031 seconds.
------------------------------------------------------------------
------------
```