Chapter 8:
Creating Crowd System

Topics:

Setting object direction, rigid body, rigid solver, and Maya expression

Chapter 8:
Creating Crowd System

เอกสารประกอบการเรียน
รายวิชา ANI 951301
สาขาวิชาแอนนิเมชันและเกม
วิทยาลัยศิลปะ สื่อ และเทคโนโลยี
มหาวิทยาลัยเชียงใหม่

951301: Advanced Animation and Rendering Techniques

วัตถุประสงค์

1. เข้าใจถึงหลักการสร้างพฤติกรรมให้กับฝูงชน (crowd simulation) ผ่านการทำงานของ MEL scripting language
2. เข้าใจถึงการสร้างและปรับทิศทางการเคลื่อนที่แบบพฤติกรรมให้กับวัตถุ
3. เรียนรู้หลักการปรับทิศทางและพฤติกรรมของวัตถุด้วย Maya expression
4. เข้าใจถึงหลักการสร้าง pseudo random ให้กับวัตถุในการเคลื่อนที่แบบไม่เป็นแบบแผน เพื่อจำลองความเสมือนจริงในการ simulation
5. นำความรู้ที่ได้มาประยุกต์ใช้สร้างการโต้ตอบระหว่างวัตถุ (interaction) ด้วยตนเอง

เนื้อหาการสอน

การกำหนดทิศทางการเคลื่อนที่ให้กับวัตถุสมมุติ การสร้างวัตถุตัวนำและตัวตาม การสร้าง rigid body และ rigid solver และการใช้ Maya expression เพื่อช่วยในการจำลองพฤติกรรมให้กับวัตถุทั้งในแบบซ้ำเป็น pattern และแบบสุ่ม

Basic Crowd Behaviour

ในการทำงานแอนนิเมชัน มีบ่อยครั้งที่เราจะต้องสร้างการเคลื่อนไหวให้กับกลุ่มของตัวละครทีละจำนวนมากๆ (crowd simulation) ที่มีการเคลื่อนไหวคล้ายๆกัน มีจุดมุ่งหมายใกล้เคียงกัน แต่การเคลื่อนไหวจะต้องไม่เหมือนกันทั้งหมด เพื่อความสมจริงเราจะต้องใส่ความเป็นเอกลักษณ์ในการเคลื่อนไหวให้กับตัวละครแต่ละตัว ซึ่งจะต้องใช้เวลาเป็นอย่างมาก ในบทนี้เราจะมาเรียนรู้การใช้ MEL scripts ในการสร้างระบบการเคลื่อนที่ของกลุ่มตัวละคร (crowd simulation system) เพื่อช่วยในการทำงานลักษณะนี้ให้มีความสะดวกสบายมากขึ้น ในการตอบโจทย์นี้เราสามารถใช้ความรู้ในส่วนของ solid body dynamic ที่เรียนมาเมื่อบทก่อนหน้านี้ และความรู้จากการจำลองพฤติกรรมให้กับกลุ่มตัวละคร (crowd behaviour) ซึ่งจะกล่าวถึงในส่วนแรกของบทนี้รวมกัน จุดประสงค์เพื่อให้นักศึกษามีความเข้าใจถึงพื้นฐานในการใส่พฤติกรรมให้กับตัวละครและสามารถนำไปประยุกต์ใช้สร้างงานที่มีความซับซ้อนเพิ่มขึ้นได้ต่อไปด้วยตนเอง

เพื่อสร้างพื้นฐานความเข้าใจในการจำลองพฤติกรรมตัวละครจำนวนมาก เราจะเริ่มจากการใส่พฤติกรรมให้กับตัวละครแบบง่ายๆทีละตัวละครก่อน ตามด้วยการสร้างความสัมพันธ์ของพฤติกรรมตัวละครที่มีต่อกัน (interaction) ก่อนที่จะเพิ่มความซับซ้อนสู่การจำลองฝูงชนขนาดใหญ่

Setting Object Direction

ในส่วนนี้เราจะเริ่มจากการเรียนรู้การสร้างวัตถุแบบง่ายๆ และทดลองกำหนดการเคลื่อนที่ให้กับวัตถุนั้นแบบเบื้องต้น ขั้นแรกเราจะสร้างวัตถุเพื่อใช้ในการทดลองนี้กันก่อน โดยเราจะใช้ polygon cube เสมือนเป็นรถที่เราต้องการให้เคลื่อนที่ ดังนั้นขั้นแรกให้สร้าง polygon cube ขึ้นมาให้มีชื่อว่า vehicle_1 และมีสัดส่วนตาม scripts ด้านล่าง

```
polyCube -name vehicle_1 -width 2 -height 2.5 -depth 2;
```

เพื่อกำหนด dynamic ให้กับวัตถุนี้ของเรา เราจะต้องสร้าง rigid body ขึ้นมาและกำการ connect เข้าไปกับวัตถุของเรา (vehicle_1) ก่อน โดยให้ตั้งชื่อว่า rigidVehicle_1 ประโยชน์ในการสร้าง rigid body นั้น คือเราสามารถกำหนดค่า dynamic ต่างๆเข้าไปได้เช่น ค่าเริ่มต้นตำแหน่งของวัตถุ (initial position) และ ค่าแรงขับเคลื่อน (impulse) ที่จะชักนำวัตถุไปในทิศทางที่ต้องการได้ ในการสร้าง rigid body สามารถทำได้ดังตัวอย่างถัดไป

```
rigidBody -name rigidVehicle_1 -active -mass 1 -bounciness
0 // สร้าง rigid body ชื่อ rigidVehicle_1
-damping 1.5 -position -10 0 0
// ค่าสูญเสียพลังงานจาก frictional force และ ตำแหน่งเริ่มต้นที่ X = -10
-impulse 0.15 0.0 0.0 vehicle_1;
// กำหนดให้วัตถุถูกแรงขับเคลื่อนไปในทิศทาง X = 0.15
```

จากคำสั่งด้านบนโปรแกรมจะทำการสร้าง rigid body ให้กับวัตถุที่ชื่อ vehicle_1 โดยให้มีค่าแรงขับเคลื่อนเป็น 0.15 และอยู่ในตำแหน่งเริ่มต้นที่ X, Y, Z = -10, 0, 0 เราจะสังเกตเห็นข้อความว่า // Result: rigidSolver // ปรากฏขึ้นที่ feedback area แปลว่าถูกต้องแล้ว rigid solver ที่ถูกสร้างขึ้นนี้จะถูก connected เข้ากับ vehicle_1 โดนอัตโนมัติ ซึ่งเราจะต้องทำการ connect วัตถุอื่นๆ ที่เราต้องการให้ dynamic มีการ interact กับ vehicle_1 เข้ากับมันด้วยตนเองในภายหลัง ในขณะนี้ให้เราปล่อยไว้แบบนี้ก่อน ถ้าเราลองกด play ตรง time control ดูจะพบว่าวัตถุเคลื่อนที่ไปตามแนวแรงที่เราตั้งโดยไม่ต้องมีการ set keyframe แต่อย่างใด

เพื่อความสะดวกในการทำงาน ขั้นต่อไปเราจะสั่งให้โปรแกรมแสดงทิศทางการเคลื่อนที่ของวัตถุด้วยหัวลูกศร สามารถทำได้โดยเพิ่มคำสั่งด้านล่างต่อท้าย script ของเราเข้าไป

```
      setAttr rigidSolver.displayVelocity 1;
```

เมื่อลอง play animation ใหม่ จะพบว่ามีลูกศรชี้จากวัตถุเพื่อบอกทิศทางการเคลื่อนที่ของมันดังภาพตัวอย่างด้านล่าง

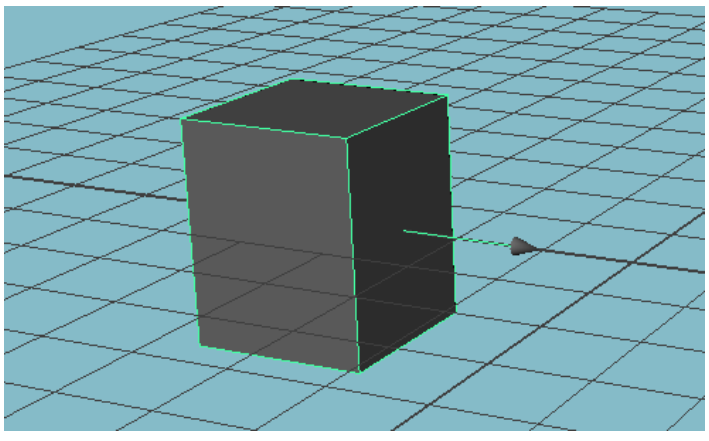

Fig 08-01: แสดงวัตถุและหัวลูกศรบอกทิศทางของแรงกระทำที่ดึงวัตถุไปข้างหน้า

เราสามารถเพิ่มหรือลดขนาดของค่า impulse ได้ด้วยคำสั่ง scaleVelocity ตามด้วยเลขจากศูนย์ถึงหนึ่ง เพื่อปรับการเคลื่อนที่ของวัตถุให้เป็นไปตามต้องการ เช่นถ้าเราต้องการจะให้แรง impulse มีค่าเป็น 80 เปอร์เซนต์ของที่ตั้งไว้เราสามารถทำได้ดังคำสั่งถัดไป

951301: Advanced Animation and Rendering Techniques

```
setAttr rigidSolver.scaleVelocity 0.8;
```

เนื่องจากจำนวนเฟรมที่เรามีตอนนี้มีความยาวไม่เหมาะสมกับการตรวจสอบการเคลื่อนไหวของวัตถุ เราควรเพิ่มจำนวนเฟรมให้เป็น 300 เฟรม ด้วยคำสั่ง playbackOption ตามตัวอย่างด้านล่าง

```
playbackOptions -min 1 -max 300;
```

จากนั้นให้ลอง execute คำสั่ง แล้วลองสังเกตการเคลื่อนที่ของวัตถุว่ามีพฤติกรรมอย่างไร จากนั้นให้ทดลองปรับเปลี่ยนค่า velocity ต่างๆดูว่ามีผลลัพธ์ต่อพฤติกรรมของวัตถุอย่างไรบ้าง

เราจะพบว่าเรามีการกำหนดทิศทางการเคลื่อนที่ให้กับวัตถุเพียงทิศทางเดียว จะเกิดอะไรขึ้นถ้าเราเพิ่มค่าแรงขับเคลื่อนไปในทิศทางอื่นด้วย ให้ทดลองเพิ่มคำสั่งด้านล่างเข้าไปต่อท้ายคำสั่งที่มีจากนั้นลองดูผลลัพธ์ที่เกิดขึ้น

```
setAttr rigidVehicle_1.impulseZ -0.15;
```

เราจะพบว่าเมื่อเพิ่มแรงขับในแกน Z เข้าไป วัตถุจะเคลื่อนที่ไปในแนวแกน X และ Z พร้อมๆกัน

Maya Expression and Expression Editor

ขั้นต่อไปเราจะมาทดลองจำลองพฤติกรรมให้กับวัตถุผ่านทาง expression เพื่อให้นักศึกษาเข้าใจถึงเป้าหมายและผลลัพธ์จากการตั้งค่าต่างๆก่อนที่จะใช้ MEL scripts สั่งให้โปรแกรมทำงานต่อไป expression สามารถใช้ในการกำหนดพฤติกรรมให้กับวัตถุได้ผ่าน scripts ขนาดสั้นๆ เราสามารถใช้ expression มาช่วยอำนวยความสะดวกในการทำงานได้หลายรูปแบบ ซึ่งทุกๆคำสั่งใน expression สามารถสั่งผ่าน MEL scripts ได้ ซึ่งจะกล่าวถึงในส่วนต่อไป

กลับมาที่งานของเรา ขณะนี้เราจะพบว่าการเคลื่อนที่ของวัตถุยังเป็นแบบ static ไม่มีการเปลี่ยนแปลงอันใด เราจะมาใช้ expression ให้การเคลื่อนที่ของวัตถุเปลี่ยนแปลงตามช่วงเวลาที่กำหนด ขั้นแรกให้เราเปิดหน้าต่าง expression editor ขึ้นมาก่อน สามารถทำได้โดยการไปที่ Window > Animation Editors > Expression Editor ตามภาพตัวอย่างด้านล่าง

951301: Advanced Animation and Rendering Techniques

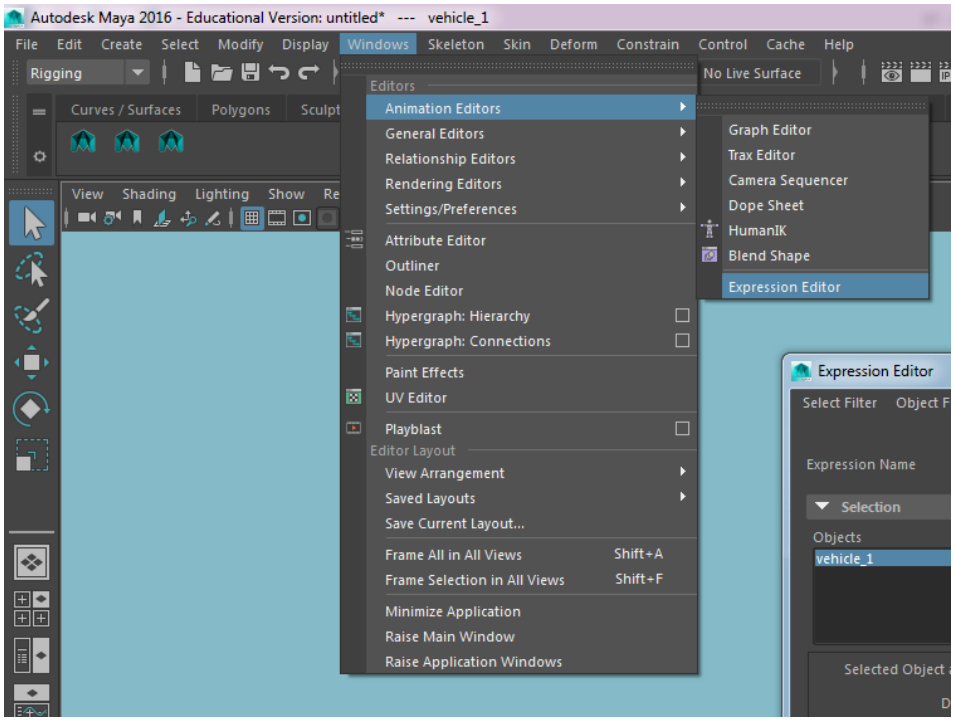

Fig 08-02: แสดงการเรียกใช้ Maya expression ในโปรแกรม Maya

เราจะได้หน้าต่าง expression ขึ้นมา ซึ่งสามารถแบ่งได้เป็นสามส่วนหลักๆดังนี้

1. Expression Name ไว้สำหรับตั้งชื่อให้กับ expression

2. Selection ซึ่งจะแบ่งออกเป็นสองส่วนคือส่วนของ Objects และส่วนของ Attributes มีไว้สำหรับเลือกวัตถุและค่า attributes ที่ต้องการดัดแปลง

3. Expression จะเป็นหน้าต่างขนาดใหญ่ มีไว้สำหรับให้พิมพ์ข้อความคำสั่งต่างๆที่ต้องการลงในส่วนนี้

ให้นักศึกษาดูให้แน่ใจว่าวัตถุ vehicle_1 ถูก selected อยู่ จากนั้นให้พิมพ์คำสั่งดังต่อไปนี้ลงในส่วนของ Expression: ด้านล่างของหน้าต่าง

```
rigidVehicle_1.impulseX = sin (time);
```

และตั้งชื่อให้กับ expression ว่า wander ตรงส่วนของ Expression Name ดังภาพตัวอย่างด้านล่าง เมื่อเสร็จแล้วให้กดปุ่ม Create ตรงมุมซ้ายสุดด้านล่างเป็นการสิ้นสุดขั้นตอนการสร้าง

951301: Advanced Animation and Rendering Techniques

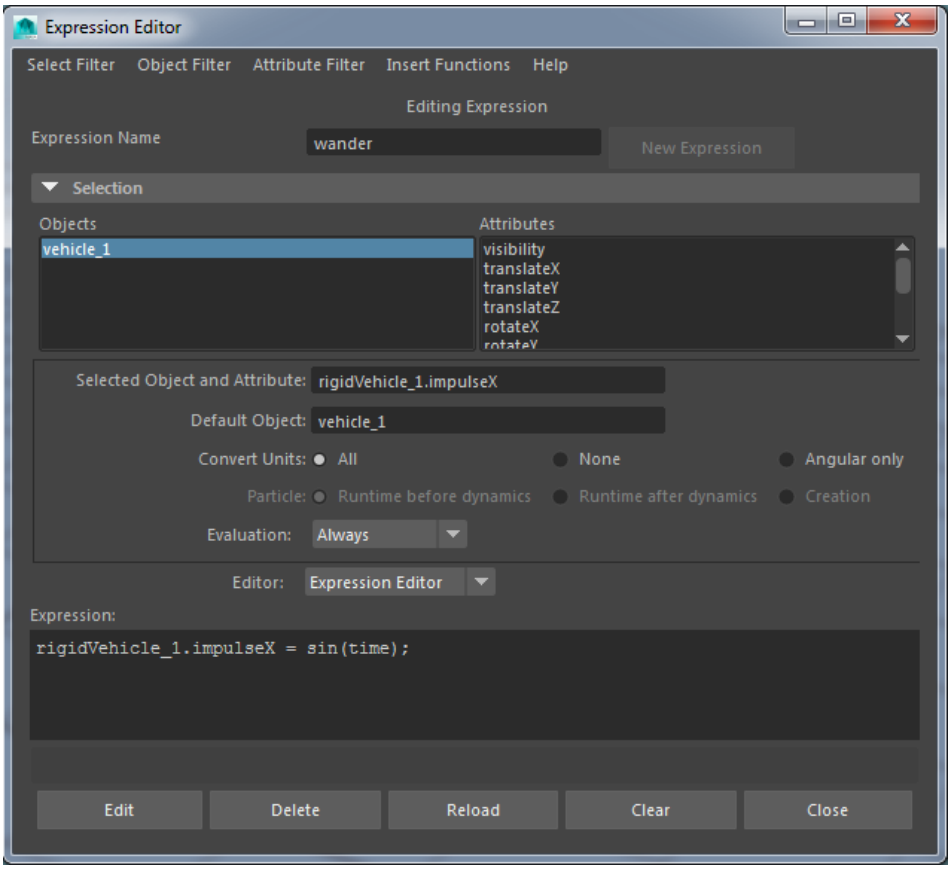

Fig 08-03: แสดงตัวอย่างของหน้าต่างหารทำงาน Expression Editor

จากนั้นให้ทดลอง play animation สังเกตจาก top view จะพบว่า vehicle_1 ของเรามีการเคลื่อนที่ไป-กลับลักษณะเป็นรูปตัว W ไปเรื่อยๆ ตามแนวแกน X ทั้งนี้เนื่องจากเรา apply ค่า sine แปรผันตามเวลาเข้าไป ทำให้วัตถุมีการเคลื่อนที่ขึ้นลงในลักษณะของคลื่น เมื่อเราได้ผลลัพธ์ตามนี้แล้ว เรามาทดลองปรับแต่งค่าอื่นๆดูเพื่อสังเกตพฤติกรรมของวัตถุว่าเป็นอย่างไร โดยเริ่มจากการทดลองใส่ค่า cos ในแนวแกน Z เพิ่มเข้าไปตามตัวอย่างด้านล่าง (ต่อจากคำสั่งเดิม)

```
rigidVehicle_1.impulseZ = cos (time);
```

จากนั้นให้กดปุ่ม Edit เนื่องจากเราจะไม่สามารถกดปุ่ม Create ได้หลังจากที่ได้ created คำสั่งไปแล้ว สังเกตพฤติกรรมของวัตถุจาก top view จะพบว่าวัตถุเคลื่อนที่เป็นลักษณะรูปตัว O ซ้ำไปเรื่อยๆเนื่องจากค่า sine และ cos ที่ใส่ไปในแนวแกน X และ Z นั่นเอง ขั้นต่อไปเราจะทดลองเพิ่มค่าเวลาของ cos ในแนวแกน Z เป็นสองเท่า เพื่อเร่งให้วัตถุเคลื่อนที่วกกลับก่อนที่จะ

951301: Advanced Animation and Rendering Techniques

ครบรอบเป็นวงกลม โดยให้นักศึกษาลบคำสั่งเดิมออกแล้วใส่คำสั่งตามตัวอย่างด้านล่างเข้าไปแทน

```
rigidVehicle_1.impulseX = sin (time);
rigidVehicle_1.impulseZ = cos (time*2);
```

จากนั้นกดปุ่ม Edit แล้วสังเกตการณ์เคลื่อนที่ของวัตถุจาก top view จะพบว่าวัตถุมีการเคลื่อนที่เป็นรูปเลข 8 ซ้ำไปเรื่อยๆ

Applying Pseudo Random to the Objects

ถึงตอนนี้นักศึกษาน่าจะมีความเข้าใจในการทำงานของ expression มากขึ้นแล้ว ที่ผ่านมาเราสร้างพฤติกรรมให้กับวัตถุในลักษณะของ patterns ซึ่งจะมีการเคลื่อนที่ทางคณิตศาสตร์แบบคาดเดาได้ ในส่วนนี้เราจะมาเรียนรู้การสร้าง pseudo random ให้กับวัตถุ pseudo random คือการสั่งให้วัตถุเคลื่อนที่ในแบบสุ่ม ซึ่งสามารถทำได้โดยการใช้คำสั่ง noise เติมเข้าไปในค่าที่ applied ลงใน impulse นั่นเอง

ให้นักศึกษาลบคำสั่งเดิมในหน้าต่าง Expression: ออก แล้วพิมพ์คำสั่งดังตัวอย่างด้านล่างเข้าไป

```
rigidVehicle_1.impulseX = sin (time);
rigidVehicle_1.impulseZ = noise (time);
```

กด Edit แล้วสังเกตพฤติกรรมของวัตถุจาก top view จะพบว่าค่า impulse แบบสุ่มในแนวแกน Z จะดึงวัตถุให้หลุดออกจากเส้นทางการเดินเดิม ออกเป็นลักษณะ random แบบคาดเดาไม่ได้

เพื่อความสะดวกในการทำงานในส่วนต่อไป เราสามารถประกาศตัวแปรขึ้นมาใน expression แล้ว assign ค่าที่ต้องการลงไปในตัวแปร เพื่อให้มาแปรผันกับเวลา เพื่อให้การเคลื่อนที่มีความหลากหลายยากจะคาดเดาเพิ่มขึ้นได้ ให้ลองประกาศตัวแปร $xMult และ $zMult ให้มีค่าเป็น 2.0 และ 1.5 แล้วเพิ่มเข้าไปในสมการการเคลื่อนที่ทั้งสองแกนตามตัวอย่างด้านล่าง (ลบคำสั่งเดิมออกก่อน)

```
float $xMult = 2.0;
float $zMult = 1.5;
rigidVehicle_1.impulseX = sin (time*$xMult);
rigidVehicle_1.impulseZ = (noise (time)*$zMult);
```

กด Edit แล้วสังเกตการณ์เคลื่อนที่ของวัตถุว่าเปลี่ยนแปลงอย่างไร ทดลองปรับค่าของตัวแปรทั้งสองและศึกษาผลลัพธ์ที่ได้ว่าเป็นอย่างไร

Workshop:
Objects and Interactions

การจัดวางวัตถุให้สอดคล้องกับทิศทางการเคลื่อนที่

ถ้าสังเกตการณ์เคลื่อนที่ของวัตถุที่เราทำ จะพบว่าวัตถุจะหันหน้าตรงไปในทิศทางเดียว โดยที่ไม่มีการหันไปตามทิศทางที่เคลื่อนที่ซึ่งเป็นเรื่องผิดธรรมชาติและทำให้การเคลื่อนที่ขาดความสมจริง ทั้งนี้เนื่องจากการหมุน (rotation) ของวัตถุถูก connected อยู่กับแกน X, Y, Z rotation channel ซึ่งเป็นคุณสมบัติพื้นฐานของ rigid solver ในส่วนนี้เราจะมาทำการอนุญาตให้วัตถุสามารถหันเข้าหาทิศทางที่เคลื่อนที่ได้อย่างอิสระ ซึ่งสามารถทำได้โดยการบังคับการ disconnection ให้กับ rigid solver และ ทำการ disconnect ค่า rotation Y ให้กับวัตถุของเรา ซึ่งสามารถทำได้ผ่านทาง expression และ MEL scripts ซึ่งเราจะอธิบายถึงในส่วนต่อไป

การสร้างพฤติกรรมโต้ตอบ (Interaction) ให้กับวัตถุแบบกลุ่ม (Crowd System)

เมื่อเราเข้าใจการจำลองการเคลื่อนไหวให้กับวัตถุผ่าน expression แล้ว ในส่วนนี้เราจะเริ่มต้นจากการนำ expression ที่สร้างมาใส่ไว้ใน MEL และเราจะเพิ่มเติมพฤติกรรมในการโต้ตอบให้กับวัตถุจำนวนมากกว่าหนึ่งวัตถุผ่านทาง MEL scripts

ในการสร้างการทำงานกับวัตถุแบบกลุ่ม (crowd system) สิ่งที่เราจะต้องคำนึงถึงเพิ่มขึ้นมานอกเหนือจากการเคลื่อนที่ของวัตถุแต่ละตัวแล้ว คือการกำหนดพฤติกรรมในการหลีกเลี่ยงการปะทะซึ่งกันและกัน (collisions) ให้กับวัตถุ เนื้อหาในส่วนนี้จะเริ่มปูพื้นฐานความเข้าใจในการ simulate crowd system โดยเริ่มจากการสร้างการเคลื่อนที่ให้กับวัตถุตัวแรก (เช่นเดียวกับเนื้อหาในส่วนที่ผ่านมา) กำหนดให้เป็น leader และการกำหนดการเคลื่อนที่ให้กับวัตถุที่สองให้คล้อยตามวัตถุแรกในลักษณะของ follower ในส่วนต่อมา

ก่อนอื่นให้เราเปิด scene ใหม่ขึ้นมา สามารถทำได้ด้วยคำสั่ง

```
file -force -new;
```


951301: Advanced Animation and Rendering Techniques

ขั้นต่อไปเราจะทำการแก้ไข expression scripts ที่เราสร้างไว้ในส่วนแรกมาใช้ใน MEL และเพิ่มเติมส่วนของการ disconnect rotation เพื่อให้วัตถุหันหน้าตามทิศทางที่เคลื่อนที่ สามารถทำได้ตามคำสั่ง MEL scripts ด้านล่าง

```
// สร้างตัว master crowd solver ไว้สำหรับ crowd ทุกตัว
rigidSolver -create -current -name crowdSolver // สร้าง rigid
solver ชื่อ crowdSolver
-velocityVectorScale 0.5 // กำหนดแรงของการเคลื่อนที่
-displayVelocity on; // ให้แสดงผลหัวลูกศรแสดงทิศทางการเคลื่อนที่
setAttr crowdSolver.allowDisconnection 1; // เพื่ออนุญาตให้วัตถุหันหน้า
ตามทิศทางที่เคลื่อน

// สร้างวัตถุเพื่อใช้ในการ simulation เสมือนเป็นรถ
polyCube -name vehicle_1 -width 2 -height 2.5 -depth 2; //
สร้างรถจำลองแบบง่ายๆ
playbackOptions -min 1 -max 300; // กำหนดเพิ่มเวลา playback time
เป็น 300 frames

// สร้าง rigid body ให้กับรถ
rigidBody -name rigidVehicle_1 -active -mass 1 - bounciness
0 // สร้าง rigid body ชื่อ rigidVehicle_1
-damping 1.5 -position -10 0 0 -impulse 0.0 0.0 0.0 //
กำหนดการสูญเสียแรง, ตำแหน่ง และแรงขับเคลื่อน
-solver crowdSolver vehicle_1; // มีผลกับ crowd solver ของ
vehicle_1

// สั่ง disconnect rotation ให้กับ rotation ทั้งสามแกนของรถ
disconnectAttr rigidVehicle_1ry.output vehicle_1.rotateY;
// อนุญาต rotation ตามแนวแกน Y
disconnectAttr rigidVehicle_1rx.output vehicle_1.rotateX;
// อนุญาต rotation ตามแนวแกน X
disconnectAttr rigidVehicle_1rz.output vehicle_1.rotateZ;
// อนุญาต rotation ตามแนวแกน Z

// สร้าง expression ให้กับรถ โดยให้ใช้ตัวแปร $expString โดยใช้คำสั่งเดิมที่เราสร้างไว้ตอนแรก
$expString = "float $xMult = 2.0; \n"; // กำหนดตัวแปรที่หนึ่งเพื่อใช้ในการ
คำนวณการเคลื่อนที่
$expString += "float $zMult = 1.5; \n"; // กำหนดตัวแปรที่สองเพื่อใช้ในการ
คำนวณการเคลื่อนที่
$expString += "rigidVehicle_1.impulseX = sin(time *
$xMult); \n"; // กำหนดการเคลื่อนที่จากค่า sine/time
$expString += "rigidVehicle_1.impulseZ = (noise(time) *
$zMult); \n \n"; //กำหนด noise เพื่อขจัด pattern

// กำหนดทิศทางการหมุนของวัตถุให้สอดคล้องกับทิศทางที่เคลื่อนที่
$expString += "float $fVel[] = `getAttr
rigidVehicle_1.velocity`; \n \n";
$expString += "vehicle_1.rotateX = 0; \n";
$expString += "vehicle_1.rotateY = atan2d($fVel[0],
$fVel[2]); \n";
$expString += "vehicle_1.rotateZ = 0; \n";
```


951301: Advanced Animation and Rendering Techniques

```
// ตั้งชื่อให้กับ expression ว่า wander และ convert unit ทั้งหมด
expression -s $expString -name wander -alwaysEvaluate true
-unitConversion all;
```

เมื่อพิมพ์คำสั่งทั้งหมดลงใน script editor แล้วทดลอง execute คำสั่ง เราจะสามารถสร้างพฤติกรรมการเคลื่อนที่ให้กับรถได้โดยไม่ต้องแยกสั่งใน expression window แต่อย่างใด ให้สังเกตการเคลื่อนที่ว่าเป็นอย่างไร สอดคล้องกับความต้องการของเราหรือไม่

การเพิ่มเติมวัตถุที่สอง

ในตอนนี้เรามีความเข้าใจในการสร้างวัตถุให้เคลื่อนที่ด้วย MEL scripts กันแล้ว ขั้นต่อไปคือการเพิ่มวัตถุที่สองเข้าไปในลักษณะเป็น follower ของวัตถุตัวแรก (วัตถุที่สองจะวิ่งเข้าหาวัตถุ leader) ในการสร้างวัตถุมากกว่าหนึ่งชิ้น สิ่งที่เราต้องระวังคือการป้องกันชนกันของวัตถุ สามารถทำได้โดยการบังคับให้วัตถุรักษาระยะห่างระหว่างกันไว้โดยการสร้าง force field ด้วยคำสั่ง radial และกำหนดขนาดและแรงของ filed ให้เหมาะสม ในส่วนของ leader เราจะต้องสร้าง global force field ขึ้นมา ซึ่งจะมีผลในการดึงดูด (attract) solid body ที่อยู่ใกล้ๆ ให้วิ่งเข้าหา โดยเราจะไม่กำหนดระยะสูงสุด (maximum distance) ของ attraction force ทำให้ในที่นี้มันจะส่งผลกับ solid body ทุกๆตัวภายในฉาก

ในขั้นตอนแรกให้เปิด scene ขึ้นมาใหม่ด้วยคำสั่ง

```
file –force –new;
```

จากนั้นให้พิมพ์คำสั่งตามตัวอย่างด้านล่างเพื่อสร้างตัว follower ขึ้นมา

```
// สร้างตัว master crowd solver ไว้สำหรับ crowd ทุกตัว
rigidSolver -create -current -name crowdSolver

// สร้าง rigid solver ชื่อ crowdSolver
-velocityVectorScale 0.5 // กำหนดแรงของการเคลื่อนที่
-displayVelocity on; // ให้แสดงผลหัวลูกศรแสดงทิศทางการเคลื่อนที่
setAttr crowdSolver.allowDisconnection 1;
// เพื่ออนุญาตให้วัตถุหันหน้าตามทิศทางที่เคลื่อน

// สร้างวัตถุขึ้นมาให้เป็น follower
polyCube -name vehicleF_1 -width 2 -height 2.5 -depth 2;
// สร้างรถจำลองแบบง่ายๆ เป็น follower

// สร้าง force field ให้กับ follower เพื่อป้องกันการชน
radial -position 0 0 0 -name vehicleFForce_1 -magnitude 50
// สร้าง radial ชื่อ vehicleFForce_1
-attenuation 0.3 -maxDistance 8.0; // กำหนดระยะใกล้สุดที่สามารถเข้าใกล้ได้
```


951301: Advanced Animation and Rendering Techniques

```
parent vehicleFForce_1 vehicleF_1;
// นำ force field ที่สร้างไปผูกติดไว้กับ vehicleF_1
playbackOptions -min 1 -max 300;
// ปรับ playback time ให้เป็น 300 frames

// สร้าง rigid body ให้กับ follower
rigidBody -name rigidVehicleF_1 -active -mass 1 -
bounciness 0 // สร้าง rigid body ชื่อ rigidVehicleF_1
-damping 1.5 -position -10 0 0 -impulse 0.15 0.0 0.0 -
standInObject cube // กำหนดค่าแรงกระทำ
-solver crowdSolver vehicleF_1;
// มีผลกับ crowd solver ของ vehicleF_1

// สั่ง disconnect rotation ให้กับ rotation ทั้งสามแกนของ follower
disconnectAttr rigidVehicleF_1ry.output vehicleF_1.rotateY;
// อนุญาต rotation ตามแนวแกน Y
disconnectAttr rigidVehicleF_1rx.output vehicleF_1.rotateX;
// อนุญาต rotation ตามแนวแกน X
disconnectAttr rigidVehicleF_1rz.output vehicleF_1.rotateZ;
// อนุญาต rotation ตามแนวแกน Z

// สร้าง expression ให้กับ vehicleF_1
$expString = "float $xMult1 = 2.0;\n";
// กำหนดตัวแปรที่หนึ่งเพื่อใช้ในการคำนวณการเคลื่อนที่
$expString += "float $zMult1 = 1.5;\n";
// กำหนดตัวแปรที่สองเพื่อใช้ในการคำนวณการเคลื่อนที่
$expString += "rigidVehicleF_1.impulseX =
sin(time*$xMult1);\n";  // กำหนดการเคลื่อนที่จากค่า sine/time
$expString += "rigidVehicleF_1.impulseZ = (noise(time) *
$zMult1);\n\n"; //กำหนด noise เพื่อขจัด pattern

// กำหนดทิศทางการหมุนของวัตถุให้สอดคล้องกับทิศทางที่เคลื่อนที่
$expString += "float $fVel1[] = `getAttr
rigidVehicleF_1.velocity`;\n\n";
$expString += "vehicleF_1.rotateX = 0;\n";
$expString += "vehicleF_1.rotateY = atan2d($fVel1[0],
$fVel1[2]);\n";
$expString += "vehicleF_1.rotateZ = 0;\n";

// ตั้งชื่อให้กับ expression ว่า wander และ convert unit ทั้งหมด
expression -s $expString -name wander -alwaysEvaluate true
-unitConversion all;
// เสร็จสิ้นการสร้าง follower
```

จากนั้นสร้างตัว leader ขึ้นมาด้วยวิธีเดียวกัน

```
// สร้างวัตถุขึ้นมาให้เป็น leader
// สร้างรถจำลองแบบง่ายๆ เป็น leader ตั้งชื่อว่า vehicleL_1
polyCube -name vehicleL_1 -width 2 -height 2.5 -depth 2;

// สร้าง force field ให้กับ leader เพื่อป้องกันการชน
radial -position 0 0 0 -name vehicleLForce_1 -magnitude 50
// สร้าง radial ชื่อ vehicleLForce_1
```


951301: Advanced Animation and Rendering Techniques

```
-attenuation 0.3 -maxDistance 8.0;
// กำหนดระยะใกล้สุดที่สามารถเข้าใกล้ได้เช่นเดียวกับ follower
parent vehicleLForce_1 vehicleL_1;
// นำ force field ที่สร้างไปผูกติดไว้กับ vehicleL_1

// สร้าง leader field
radial -position 0 0 0 -name vehicleLeadGlobalForce_1
// สร้าง radial ชื่อ vehicleLeadGlobalForce_1
-magnitude -1 -attenuation 1.2;
// กำหนดขนาดของ filed และ การสูญเสียค่า
parent vehicleLeadGlobalForce_1 vehicleL_1;
// นำ force field ที่สร้างไปผูกติดไว้กับ vehicleL_1

// สร้าง rigid body ให้กับ leader
rigidBody -name rigidVehicleL_1 -active -mass 1 -
bounciness 0 // สร้าง rigid body ชื่อ rigidVehicleL_1
-damping 1.5 -position 10 0 0 -impulse 0.15 0.0 0.0
// กำหนดค่าแรงกระทำ
-standInObject cube -solver crowdSolver vehicleL_1;
// มีผลกับ crowd solver ของ vehicleL_1

// สั่ง disconnect rotation ให้กับ rotation ทั้งสามแกนของ leader
disconnectAttr rigidVehicleL_1ry.output vehicleL_1.rotateY;
disconnectAttr rigidVehicleL_1rx.output vehicleL_1.rotateX;
disconnectAttr rigidVehicleL_1rz.output vehicleL_1.rotateZ;
```

สร้างพฤติกรรมให้กับ leader

```
// สร้าง expression ให้กับ vehicleL_1
$expString = "float $xMult2 = -2.0; \n";
$expString += "float $zMult2 = 2.5; \n";
$expString += "rigidVehicleL_1.impulseX = sin(time *
$xMult2); \n";
$expString += "rigidVehicleL_1.impulseZ = (noise(time) *
$zMult2); \n\n";

// กำหนดทิศทางการหมุนของวัตถุให้สอดคล้องกับทิศทางที่เคลื่อนที่
$expString += "float $fVel2[]; \n";
$expString += "$fVel2 = `getAttr rigidVehicleL_1.velocity`;
\n\n";
$expString += "vehicleL_1.rotateX = 0; \n";
$expString += "vehicleL_1.rotateY = atan2d($fVel2[0],
$fVel2[2]); \n";
$expString += "vehicleL_1.rotateZ = 0; \n";

// ตั้งชื่อให้กับ expression ว่า wanderL และ convert unit ทั้งหมด
expression -s $expString -name wanderL_exp1 -alwaysEvaluate
true
-unitConversion all;
// เสร็จสิ้นขั้นตอนการสร้าง leader ขั้นต่อไปเราจะทำการเชื่อมต่อความสัมพันธ์ระหว่าง leader และ
follower เข้าด้วยกัน
// ทำการเชื่อมความสัมพันธ์ของ dynamic force field ของ leader เข้ากับ rigid
body ของ follower
```


951301: Advanced Animation and Rendering Techniques

```
connectDynamic -fields vehicleLForce_1 rigidVehicleF_1;
// เชื่อม dynamic global force field ของ leader เข้ากับ rigid body
ของ follower
connectDynamic -fields vehicleLeadGlobalForce_1
rigidVehicleF_1;
// เชื่อมความสัมพันธ์ของ dynamic force field ของ follower เข้ากับ rigid
body ของ leader
connectDynamic -fields vehicleFForce_1 rigidVehicleL_1;
```

เมื่อเสร็จแล้วให้ execute คำสั่ง จะพบว่า leader วิ่งไปในทิศทางที่ยากจะคาดเดา ในขณะที่ follower พยายามจะวิ่งตาม ทั้งๆที่ทั้งคู่ต่างมี expression หรือพฤติกรรมเป็นของตนเอง

Reference

- Wilkins, M. R. and Kazmier, C. (2005) *MEL Scripting for Maya Animations*, Morgan Kaufmann Publishers, Elsevier Inc.
- Galanakis, R. (2014) *Practical Maya Programming with Python*, Packt Publishing Ltd.
- Stripinis, D. (2003) *The MEL Companion: Maya Scripting 3D Artists*, Charles River Media, INC