การใช้เซลลูลาร์ออโตมาตาในการลดการคำนวณในปัญญาประดิษฐ์แบบใช้เอเจนต์สำหรับการทำฟลอกกิง

นาย กิติคุณ จงเสรีกิจ

วิทยานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิทยาศาสตรมหาบัณฑิต
สาขาวิชาวิทยาศาสตร์คอมพิวเตอร์ ภาควิชาวิศวกรรมคอมพิวเตอร์
คณะวิศวกรรมศาสตร์ จุฬาลงกรณ์มหาวิทยาลัย
ปีการศึกษา 2552

USING CELLULAR AUTOMATA TO REDUCE CALCULATIONS IN AGENT- BASED FLOCKING ARTIFICIAL INTELLIGENCE

Mr. Kitikun Jongsarikit

A Thesis Submitted in Partial Fulfillment of the Requirements
for the Degree of Master of Science Program in Computer Science

Department of Computer Engineering

Faculty of Engineering

Chulalongkorn University

Academic Year 2009

หัวข้อวิทยานิพนธ์	การใช้เซลลูลาร์ออโตมาตาในการลดการคำนวณในปัญญาประดิษฐ์แบบใช้เอเจนต์สำหรับการทำฟลอกกิง
โดย	นายกิติคุณ จงเสรีกิจ
สาขาวิชา	วิทยาศาสตร์คอมพิวเตอร์
อาจารย์ที่ปรึกษาวิทยานิพนธ์หลัก	ผู้ช่วยศาสตราจารย์ ดร.วิษณุ โคตรจรัส

คณะวิศวกรรมศาสตร์ จุฬาลงกรณ์มหาวิทยาลัย อนุมัติให้นับวิทยานิพนธ์ฉบับนี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญามหาบัณฑิต

..... คณบดีคณะวิศวกรรมศาสตร์

(รองศาสตราจารย์ ดร.บุญสม เลิศหิรัญวงศ์)

คณะกรรมการสอบวิทยานิพนธ์

..... ประธานกรรมการ

(ผู้ช่วยศาสตราจารย์ ดร.อดิวงศ์ สุชาโต)

..... อาจารย์ที่ปรึกษาวิทยานิพนธ์หลัก

(ผู้ช่วยศาสตราจารย์ ดร.วิษณุ โคตรจรัส)

..... กรรมการ

(ผู้ช่วยศาสตราจารย์ ดร.เศรษฐา ปานงาม)

..... กรรมการภายนอกมหาวิทยาลัย

(ผู้ช่วยศาสตราจารย์ ดร.สุรพงษ์ เลิศสิทธิชัย)

นายกิติคุณ จงเสรีกิจ : การใช้เซลลูลาร์ออโตมาตาในการลดการคำนวณในปัญญาประดิษฐ์แบบใช้เอเจนต์สำหรับการทำฟลอกกิง. (Using cellular automata to reduce calculations in agent-based flocking artificial intelligence) อ.ที่ปรึกษาวิทยานิพนธ์หลัก : ผศ.ดร.วิษณุ โคตรจรัส, 80 หน้า.

โปรแกรมจัดการอนิเมชันโดยทั่วไปใช้การควบคุมฝูงชนโดยควบคุมเอเจนต์แต่ละตัว ซึ่งการใช้เอเจนต์หนึ่งตัวแทนตัวละครหนึ่งตัวนั้นเปิดโอกาสให้ผู้ทำอนิเมชันสามารถเปลี่ยนแปลงตัวละครเฉพาะตัวได้ในรายละเอียด โดยยังคงพฤติกรรมรูปแบบมาตรฐานไว้กับตัวละครอื่นๆได้ แต่การที่ตัวละครแต่ละตัวต้องตัดสินใจนั้นทำให้การประมวลผลทำได้ช้าเพราะซีพียูต้องคำนวณการตัดสินใจของเอเจนต์แต่ละตัว ทีละตัว งานวิทยานิพนธ์นี้นำเสนอเทคนิคที่เรียกว่าเซลลูลาร์ฟลอกกิง ซึ่งเป็นการลดการทำงานของซีพียู โดยให้เอเจนต์ที่อยู่ในพื้นที่เดียวกันใช้สมองร่วมกัน ดังนั้นการตัดสินใจในระดับกลุ่มจึงเกิดขึ้นได้โดยอาศัยเทคนิคฟลอกกิงที่กระทำในระดับของเซลลูลาร์ออโตมาตา ทำให้สามารถลดการคำนวณได้อย่างมาก การรักษาระยะห่างระหว่างเอเจนต์และการคำนวณทิศทางของเอเจนต์ได้ถูกเปลี่ยนมาเป็นการคำนวณในระดับกลุ่ม ในขณะที่การหลีกเลี่ยงการชนกันนั้นไม่ถูกนำมาใช้งาน งานวิทยานิพนธ์นี้ได้สร้างโปรแกรมต้นแบบด้วยแม็กซ์สคริปต์ ผลการทดลองพบว่า วิธีการที่นำเสนอนั้น สามารถลดการคำนวณได้อย่างมากโดยที่พฤติกรรมที่แสดงออกยังเป็นที่ยอมรับได้สำหรับการเคลื่อนที่ของฝูงชนในระยะจำกัด

ภาควิชา วิศวกรรมคอมพิวเตอร์ ลายมือชื่อนิสิต กิติคุณ จงเสรีกิจ
สาขาวิชา วิทยาศาสตร์คอมพิวเตอร์ ลายมือชื่ออ.ที่ปรึกษาวิทยานิพนธ์หลัก
ปีการศึกษา 2552

4970226021 : MAJOR COMPUTER SCIENCE

KEYWORDS : CROWD ANIMATION / INTELLIGENT AGENT / FLOCKING / CELLULAR AUTOMATA

KITIKUN JONGSARIKIT : USING CELLULAR AUTOMATA TO REDUCE CALCULATION IN AGENT-BASED FLOCKING ARTIFICIAL INTELLIGENCE. THESIS ADVISOR : ASST.PROF. VISHNU KOTRAJARAS, Ph.D., 80 pp.

Commercial animation software utilizes its crowd feature based on agent technologies. Using an intelligent agent for one character allows animators to easily modify a specific character's behavior in detail, while most other characters can still use the same behavioral template. An agent based crowd, however, suffers from poor performance because a CPU needs to calculate each and every agent's decision. This thesis presents Cellular Flocking, an approach for reducing the CPU load. By giving agents in the same map cell a shared brain, a group decision can be made using flocking algorithm at cellular automata level. This reduces the calculations significantly. Maintaining the distance among agents and computing agents' direction are made into group decisions, while collision avoidance is omitted. The prototype was implemented in Max Script. Results show that the proposed technique not only significantly reduces the calculations, but also maintains acceptable group movement in a limited distance.

Department : Computer Engineering Student's Signature Kitikun Jongsarikit

Field of Study : Computer Science Advisor's Signature

Academic Year : 2009

กิตติกรรมประกาศ

ขอขอบพระคุณ บิดา มารดา และครอบครัว ที่เป็นกำลังใจสำคัญ และให้ความช่วยเหลือในทุกๆด้าน จนผู้วิจัยสามารถทำวิทยานิพนธ์ฉบับนี้เสร็จ

ขอขอบพระคุณอาจารย์ที่ปรึกษา ผศ.ดร. วิษณุ โคตรจรัส ซึ่งเป็นผู้มอบโอกาส อีกทั้งยังให้คำปรึกษา และคอยแนะนำแนวทางในการทำวิจัย ตลอดจนเป็นผู้ตรวจทานและแก้ไข จนทำให้วิทยานิพนธ์ฉบับนี้สำเร็จลุล่วง

ขอขอบพระคุณคณะกรรมการสอบวิทยานิพนธ์ ผศ.ดร. อติวงศ์ สุชาโต ผศ.ดร. โชติรัตน์ รัตนามหัทธนะ ผศ.ดร. เศรษฐา ปานงาม และ ผศ.ดร.สุรพงษ์ เลิศสิทธิชัย ที่ได้กรุณาในคำแนะนำในการแก้ไขวิทยานิพนธ์ให้มีคุณภาพมากยิ่งขึ้น วิทยานิพนธ์ฉบับนี้ไม่อาจสำเร็จได้หากไม่ได้รับความร่วมมือจากทุกท่าน และขอขอบคุณเพื่อนๆที่คอยช่วยเหลือและแนะนำวิธีการแก้ไขปัญหาต่างๆ รวมทั้งช่วยเป็นคนทดสอบคุณภาพของโปรแกรมที่ได้ทำขึ้นมา

สารบัญ

หน้า

บทคัดย่อภาษาไทย..... ง

บทคัดย่อภาษาอังกฤษ..... จ

กิตติกรรมประกาศ..... ฉ

สารบัญ..... ช

สารบัญตาราง..... ฌ

สารบัญภาพ..... ญ

บทที่ 1 บทนำ..... 1

1.1 ความเป็นมาและความสำคัญของปัญหา..... 1

1.2 วัตถุประสงค์ของการวิจัย..... 2

1.3 ขอบเขตของการวิจัย..... 2

1.4 คำจำกัดความที่ใช้ในการวิจัย..... 3

1.5 ประโยชน์ที่คาดว่าจะได้รับ..... 3

1.6 วิธีดำเนินการวิจัย..... 3

1.7 ลำดับขั้นตอนในการเสนอผลการวิจัย..... 4

บทที่ 2 เอกสารและงานวิจัยที่เกี่ยวข้อง..... 5

2.1 แนวคิดและทฤษฎี..... 5

2.1.1 เซลลูลาร์ออโตมาตา..... 5

2.1.2 อัลกอริทึมฟลอกกิง..... 6

2.1.3 มาสซีพ..... 8

2.2 งานวิจัยที่เกี่ยวข้อง..... 9

บทที่ 3 วิธีดำเนินการวิจัย..... 18

3.1 การทำงานโดยรวมของโปรแกรม..... 18

3.2 การทำงานของเซลลูลาร์ออโตมาตา..... 19

3.3 การชนกันของเอเจนต์..... 25

บทที่ 4 การทดลองและผลการทดลอง..... 27

4.1 วิธีการทดลอง..... 27

4.2 การทำงานของสถานการณ์ที่ใช้อัลกอริธึมฟลอกกิแบบปกติ..... 28

4.3 สถานการณ์จำลองจากภพยนตร์..... 31

4.4 ผลการทดลอง..... 36

หน้า

4.4.1 ผลการทดลองจากโปรแกรมเซลลูลาร์ฟลอกกิง..... 36
4.4.2 ผลการทดลองจากโปรแกรมมัลติเอเจนต์ฟลอกกิงที่ใช้อัลกอริทึมแบบธรรมดา..... 47
4.5 ผลการใช้ทรัพยากรของคอมพิวเตอร์..... 56
บทที่ 5 สรุปผลการทดลอง อภิปรายผล และข้อเสนอแนะ..... 64
5.1 สรุปผลการทดลองและอภิปราย..... 64
5.2 ข้อเสนอแนะ..... 65
5.2.1 การปรับปรุงในส่วนของเซลลูลาร์ฟลอกกิง..... 65
5.2.2 การปรับปรุงในส่วนของมัลติเอเจนต์ฟลอกกิง..... 65
รายการอ้างอิง..... 66
ภาคผนวก..... 68
ภาคผนวก ก ผลงานการตีพิมพ์..... 69
ภาคผนวก ข หน้าจอผู้ใช้งาน..... 78
ประวัติผู้เขียนวิทยานิพนธ์..... 80

สารบัญตาราง

ตารางที่		หน้า
1	แสดงการใช้ทรัพยากรของเครื่องของโปรแกรมเซลลูลาร์ฟลอกกิง.....	57
2	แสดงการใช้ทรัพยากรของเครื่องของโปรแกรมมัลติเอเจนต์ฟลอกกิง.....	58
3	แสดงการเปรียบการใช้ทรัพยากรของทั้ง 2 โปรแกรม.....	60
4	แสดงผลคะแนนจากทำแบบสอบถาม.....	63

สารบัญภาพ

ภาพที่		หน้า
1	ภาพจากภาพยนตร์เรื่องอภินิหารตำนานแห่งนาร์เนีย ตอน ราชสีห์ แม่มด กับตู้พิศวง……..	2
2	แสดงเซลลูลาร์ออโตมาตาแบบ 1 มิติ ของRule 30 ที่มีการประมวลผล 15 ครั้ง….	6
3	แสดงรูปแบบของเซลลูลาร์ออโตมาตาแบบ 2 มิติพร้อมเลรอบข้าง……..	6
4	แสดงกฎของการควบคุมฝูงเอเจนต์ด้วยอัลกอริทึมฟลอกกิง……	7
5	แสดงการวิ่งอย่างอลหม่านของฝูงวิลเดอบีสในภาพยนตร์เรื่องไลออนคิง……	8
6	แสดงหน้าต่างการทำงานของมาสซีพซอฟแวร์……	9
7	แสดการรวมกลุ่มรูปสี่เหลี่ยมและสามเหลี่ยม……..	11
8	แสดงความสัมพันธ์ของเลเยอร์ทั้ง 2 ขั้น……	12
9	แสดงหน้าจอแสดงผลการเดินของเอเจนต์โดยเซลลูลาร์ออโตมาตา……	12
10	แสดงโครงสร้างของสนามฟุตบอล……	13
11	แสดงสถานะเริ่มต้นที่ได้จากการสุ่มตำแหน่งที่ฝนตกและการระเบิด ก่อนเกิดไฟไหม้……	14
12	แสดงแบบจำลอง 2 มิติของ Flat Walk รูป a และแบบจำลอง 3 มิติของ Flat Walk ในรูป b……..	15
13	แสดงการเดินตัดกันของฝูงคน 4 กลุ่ม……..	17
14	แสดงการทำงานโดยรวมของระบบ……..	19
15	แสดงส่วนประกอบโดยรวมของระบบ.....	19
16	แสดงค่าระยะทางจากการใช้อินฟลูเอนซ์แมป……..	21
17	แสดงวิธีหาค่ามุมของทฤษฎีปีทากอรัส.....	22
18	แสดงทิศทางที่เป็นไปได้จากการคำนวณในสถานะแรก……	23
19	แสดงการหาทิศทางของเซลจากเซลรอบข้างของสมการที่ 3……	24
20	แสดงค่ามุมของเซลแต่ละเซลที่ใช้ในการคำนวน.....	25
21	แสดงรหัสเทียมของการเคลื่อนที่ของบอยโยรวม.....	29
22	แสดงรหัสเทียมของกฎข้อที่ 1……	29
23	แสดงรหัสเทียมของกฎข้อที่ 2.....	30

ภาพที่ หน้า

24 แสดงรหัสเทียมของกฎข้อที่ 3..... 30
25 แสดงภาพการเคลื่อนที่โดยรวมของสถานการณ์จำลองที่ 1..... 31
26 แสดงภาพการเคลื่อนที่โดยรวมของสถานการณ์จำลองที่ 2..... 32
27 แสดงภาพการเคลื่อนที่โดยรวมของสถานการณ์จำลองที่ 3..... 32
28 แสดงภาพการเคลื่อนที่โดยรวมของสถานการณ์จำลองที่ 4..... 33
29 แสดงภาพการเคลื่อนที่โดยรวมของสถานการณ์จำลองที่ 5..... 33
30 แสดงภาพการเคลื่อนที่โดยรวมของสถานการณ์จำลองที่ 6..... 34
31 แสดงภาพการเคลื่อนที่โดยรวมของสถานการณ์จำลองที่ 7..... 34
32 แสดงภาพการเคลื่อนที่โดยรวมของสถานการณ์จำลองที่ 8..... 35
33 แสดงภาพการเคลื่อนที่โดยรวมของสถานการณ์จำลองที่ 9..... 35
34 แสดงภาพการเคลื่อนที่โดยรวมของสถานการณ์จำลองที่ 10..... 36
35 แสดงภาพการเคลื่อนที่โดยรวมจากโปรแกรมในสถานการณ์จำลองที่ 1..... 37
36 แสดงภาพการเคลื่อนที่โดยรวมจากโปรแกรมในสถานการณ์จำลองที่ 2..... 38
37 แสดงภาพการเคลื่อนที่โดยรวมจากโปรแกรมในสถานการณ์จำลองที่ 3..... 39
38 แสดงภาพการเคลื่อนที่โดยรวมจากโปรแกรมในสถานการณ์จำลองที่ 4..... 40
39 แสดงภาพการเคลื่อนที่โดยรวมจากโปรแกรมในสถานการณ์จำลองที่ 5..... 41
40 แสดงภาพการเคลื่อนที่โดยรวมจากโปรแกรมในสถานการณ์จำลองที่ 6..... 42
41 แสดงภาพการเคลื่อนที่โดยรวมจากโปรแกรมในสถานการณ์จำลองที่ 7..... 43
42 แสดงภาพการเคลื่อนที่โดยรวมจากโปรแกรมในสถานการณ์จำลองที่ 8..... 44
43 แสดงภาพการเคลื่อนที่โดยรวมจากโปรแกรมในสถานการณ์จำลองที่ 9..... 45
44 แสดงภาพการเคลื่อนที่โดยรวมจากโปรแกรมในสถานการณ์จำลองที่ 10..... 46
45 แสดงภาพการเคลื่อนที่โดยรวมจากโปรแกรมในสถานการณ์จำลองที่ 1..... 47
46 แสดงภาพการเคลื่อนที่โดยรวมจากโปรแกรมในสถานการณ์จำลองที่ 2..... 48
47 แสดงภาพการเคลื่อนที่โดยรวมจากโปรแกรมในสถานการณ์จำลองที่ 3..... 49
48 แสดงภาพการเคลื่อนที่โดยรวมจากโปรแกรมในสถานการณ์จำลองที่ 4..... 50
49 แสดงภาพการเคลื่อนที่โดยรวมจากโปรแกรมในสถานการณ์จำลองที่ 5..... 51
50 แสดงภาพการเคลื่อนที่โดยรวมจากโปรแกรมในสถานการณ์จำลองที่ 6..... 52
51 แสดงภาพการเคลื่อนที่โดยรวมจากโปรแกรมในสถานการณ์จำลองที่ 7..... 53

ภาพที่		หน้า
52	แสดงภาพการเคลื่อนที่โดยรวมจากโปรแกรมในสถานการณ์จำลองที่ 8……..	54
53	แสดงภาพการเคลื่อนที่โดยรวมจากโปรแกรมในสถานการณ์จำลองที่ 9……..	55
54	แสดงภาพการเคลื่อนที่โดยรวมจากโปรแกรมในสถานการณ์จำลองที่ 10……	56
55	แสดงผลการเปรียบเทียบเวลาที่ใช้ในการประมวลผลของเซลลูลาร์ฟลอกกิงกับมัลติเอเจนต์ฟลอกกิง.....	59
56	แสดงการเปรียบเทียบการใช้เวลาของเอเจนต์จำนวน 100 , 200, 300 และ 400 ตัว ในสถานการณีที่ 5.....	61
57	แสดงการเปรียบเทียบการใช้เวลาของเอเจนต์จำนวน 100 , 200, 300 และ 400 ตัว ในสถานการณีที่ 7.....	61
58	แสดงการเปรียบเทียบการใช้เวลาของเอเจนต์จำนวน 100 , 200, 300 และ 400 ตัว ในสถานการณีที่ 10.....	62
59	แสดงหน้าจอผู้ใช้งานที่ใช้สร้างเอเจนต์ในตำแหน่งต่างๆ……..	78
60	แสดงหน้าจอผู้ใช้งานของโปรแกรมเซลลูลาร์ฟลอกกิง…….	79
61	แสดงหน้าจอผู้ใช้งานของโปรแกรมมัลติเอเจนต์ฟลอกกิง……	79

บทที่ 1

บทนำ

1.1 ความเป็นมาและความสำคัญของปัญหา

ในปัจจุบันการควบคุมฝูงชน โดยเฉพาะฝูงชนขนาดใหญ่ที่แต่ละคนมีจุดมุ่งหมายร่วมกัน นิยมใช้กันอย่างแพร่หลายในภาพยนตร์ เกม รวมไปถึงภาพยนตร์โฆษณา เพื่อให้ฉากที่เห็นคนจำนวนมากเกิดความยิ่งใหญ่และสมจริง ลักษณะของฝูงชนมีมากมาย เช่น ฉากเดินทัพ การประชุม หรือแม้แต่ฉากการสู้รบ ดังที่ได้พบในภาพยนตร์หลายเรื่อง ซึ่งซอฟต์แวร์ตัวที่ได้รับการยอมรับมากที่สุดในงานควบคุมฝูงชนนั้นคือ มาสซีฟ ซึ่งใช้ในงานกำกับภาพยนตร์เรื่อง นาร์เนีย[1] ดังรูปที่ 1 ซึ่งทำงานโดยให้เอเจนต์แต่ละตัวมีสมองเป็นของตัวเอง โดยสมองหลักสร้างจากฟัซซีโลจิก เอเจนต์แต่ละตัวจะมองเห็นทัศนะวิสัยรอบข้างในพิกัดสามมิติได้แบบเดียวกับมนุษย์ และสามารถได้ยินเสียงได้ ทำให้ผู้ควบคุมสามารถกำหนดกฎการเคลื่อนไหวต่างๆได้สะดวก และทำให้เอเจนต์ทำงานได้สมจริง แต่ความต้องการทรัพยากรในการคำนวณก็สูงตามไปด้วย ซึ่งส่งผลต่อต้นทุนในการผลิตอีกเช่นเดียวกัน

ในประเทศไทยนั้น ธุรกิจภาพยนตร์ อนิเมชัน รวมไปถึงเกมนั้น ผู้ประกอบการส่วนใหญ่มีขนาดเล็กกว่าบริษัทต่างชาติเป็นอย่างมาก มีต้นทุนในระดับที่ต่ำกว่าอย่างเห็นได้ชัด เป็นเหตุให้ไม่สามารถนำเครื่องมือที่ทันสมัยที่สุดแต่ต้องเสียค่าใช้จ่ายสูงอย่างมาสซีฟมาใช้ประโยชน์ได้ ภาพยนตร์อนิเมชันที่มีชื่อเสียง เช่นก้านกล้วยภาค 1 ใช้เพียงเครื่องมือวางตัวละครในฉากเท่านั้น ไม่ได้ใช้ปัญญาประดิษฐ์ในการควบคุมกลุ่มตัวละครในฉากกองทัพแต่อย่างใด [2]

ในงานวิทยานิพนธ์นี้ ผู้เขียนมีความต้องการที่จะสร้างซอฟต์แวร์สำหรับควบคุมการเคลื่อนที่ของฝูงชนที่ใช้หลักการควบคุมเอเจนต์โดยใช้ทรัพยากรในการคำนวณน้อย งานวิจัยต่างๆที่พยายามลดทรัพยากรในการคำนวณนั้น ใช้หลักการคำนวณการไหลของของเหลวเข้ามาแทนการใช้เอเจนต์ ทำให้ผู้ใช้งานที่ต้องปรับแต่งสภาพฝูงชนจำเป็นต้องทราบหลักการคำนวณการไหล ส่งผลให้การใช้งานเป็นไปได้ยากสำหรับผู้ใช้งานด้านอนิเมชันทั่วไป งานวิทยานิพนธ์นี้ต้องการเสนอวิธีการลดการคำนวณสำหรับฝูงชนที่สามารถนำไปใช้งานได้ง่ายโดยบุคคลที่ทำงานด้านอนิเมชันทั่วไป

งานวิทยานิพนธ์นี้ใช้แนวคิดการนำเซลลูลาร์ออโตมาตามาใช้ในการควบคุมกลุ่มคน งานวิจัยในอดีตที่ใช้เซลลูลาร์ออโตมาตาในการจำลองกลุ่มคนนั้น จะจำลองคนหนึ่งคนต่อหนึ่งช่องของเซลลูลาร์ออโตมาตา แต่ว่าในงานวิทยานิพนธ์นี้ จะใช้หนึ่งช่องของเซลลูลาร์ออโตมาตาต่อคนที่อยู่ในพื้นที่ของเซลทั้งหมด เอเจนต์ที่อยู่ในเซลลูลาร์ออโตมาตาเดียวกันจะใช้สมองหลักเดียวกันเท่าที่จะใช้ได้เพื่อเป็นการประหยัดทรัพยากรในการคำนวณ นอกเหนือจากนั้นเอเจนต์ในเซลลูลาร์ออโตมาตานั้นไม่จำเป็นต้องมีระบบเซ็นเซอร์ที่ละเอียดเลียนแบบมนุษย์แต่อย่างใด เนื่องจากสามารถตรวจสอบสภาพรอบข้างได้จากสถานะของเซลที่อยู่และสถานะของเซลที่ล้อมรอบ ทำให้ประหยัดทรัพยากรในการคำนวณได้เพิ่มขึ้น

รูปที่ 1 ภาพจากภาพยนตร์เรื่องอภินิหารตำนานแห่งนาร์เนีย ตอน ราชสีห์ แม่มด กับตู้พิศวง

1.2 วัตถุประสงค์ของการวิจัย

เพื่อเสนอวิธีลดการคำนวณของฝูงชนโดยใช้เซลลูลาร์ออโตมาตาแทนการคำนวณระดับเอเจนต์ แล้วทำการประเมินผล

1.3 ขอบเขตของการวิจัย

1.กำหนดเป้าหมายหรือทิศทางการเคลื่อนที่ให้กับฝูงชนโดยอาศัยอินฟลูเอนซ์แมป

2.ฝูงชนเคลื่อนที่โดยมีเป้าหมายเดียวกัน

3.การวัดผล ทำโดยสร้างสถานการณ์จำลองตามตัวอย่างจากภาพยนตร์หรือจากโฆษณาอย่างน้อย 10 ตัวอย่าง แล้วนำผลมาเปรียบเทียบกับผลจากการใช้เครื่องมือต้นแบบที่ได้สร้างขึ้นมา

4.สถานการณ์จำลองที่นำมาใช้จะเป็นการเคลื่อนที่บนพื้นที่ราบในระยะทางจำกัด (ไม่เกิน 400 ก้าวสำหรับหนึ่งตัวละคร) เท่านั้น

5.ฝูงชนแสดงการเคลื่อนที่ไปตามพื้นเท่านั้นโดยไม่ออกท่าทางการเคลื่อนไหวอย่างอื่นเช่นท่าเดินหรือวิ่ง

6.พื้นที่ที่ใช้สำหรับการทดลองสถานการณ์จำลองที่ใช้ มีขนาดที่จุตัวละครได้ 1600 ตัว

1.4 คำจำกัดความที่ใช้ในการวิจัย

ฟลอกกิง ปัญญาประดิษฐ์ของการเคลื่อนไหวกลุ่มคน เซลลูลาร์ออโตมาตา อินฟลูเอนซ์แมป

1.5 ประโยชน์ที่คาดว่าจะได้รับ

1.ได้อัลกอริทึมสำหรับการควบคุมการเคลื่อนที่ของฝูงชนในระยะทางจำกัดโดยสามารถลดการคำนวณได้มาก

2.นักวิจัยและพัฒนาเกมและภาพยนตร์สามารถนำวิธีการที่นำเสนอไปใช้ในงานจริงเพื่อลดต้นทุนในการผลิต

1.6 วิธีดำเนินการวิจัย

1. สร้างสถานการณ์ตัวอย่างขึ้นมา 10 สถานการณ์ โดยอาศัยสถานการณ์ที่เคยถูกใช้จริงในภาพยนตร์หรือโฆษณาเป็นต้นแบบ

2.แต่ละสถานการณ์นั้นทำทั้งฟลอกกิงธรรมดาและฟลอกกิงแบบประมาณที่นำเสนอในงานวิทยานิพนธ์นี้ (ปรับคุณภาพให้ได้ตามสถานการณ์) เปรียบเทียบคุณภาพว่าเป็นที่ยอมรับได้หรือไม่โดยใช้แบบสอบถาม สอบถามจากกลุ่มตัวอย่างเป็นจำนวนอย่างน้อย 10 คน และเปรียบเทียบภาระในการคำนวณในแต่ละสถานการณ์

3.สรุปผลการเปรียบเทียบแต่ละสถานการณ์ และสรุปผลงานวิจัย

1.7 ลำดับขั้นตอนในการเสนอผลการวิจัย

1.กำหนดกฎการเปลี่ยนสถานะของเซลลูลาร์ออโตมาตาใน 3ds max

2.สร้างเซลลูลาร์ออโตมาตาใน 3ds max

3.สร้างเอเจนต์

4.ทำการเชื่อมต่อเอเจนต์เข้ากับเซลลูลาร์ออโตมาตา

5.สร้างสถานการณ์จำลอง

6.ทดสอบการควบคุมเอเจนต์ของเซล 1 เซล

7.ทดลองให้กลุ่มเอเจนต์เคลื่อนไหวตามสถานการณ์ทดสอบ แล้ววัดผลโดยซักถามจากกลุ่มตัวอย่าง

8.สรุปผลการวิจัย

บทที่ 2
เอกสารและงานวิจัยที่เกี่ยวข้อง

2.1 แนวคิดและทฤษฎี

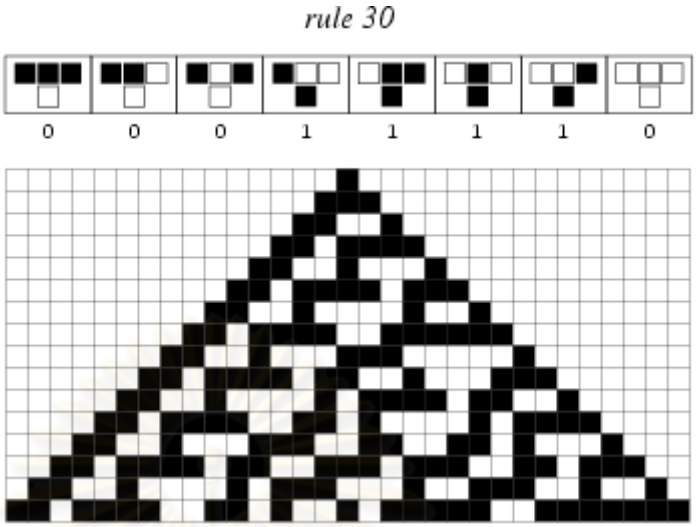
2.1.1 เซลลูลาร์ออโตมาตา

เซลลูลาร์ออโตมาตา คือ การศึกษาแบบจำลองที่ไม่ต่อเนื่องในทฤษฎีจำนวน ซึ่งประกอบด้วยกริดของเซลที่มีสถานะจำกัด กริดสามารถมีค่าเป็นไปได้ทุกตัวเลขตามจำนวนมิติ เวลาที่ใช้เป็นเวลาแบบไม่ต่อเนื่อง และสถานะของเซลคือสมการที่คำนวณสถานะของเซลที่เวลา t-1 ซึ่งค่าที่ได้ขึ้นอยู่กับเซลรอบข้างของเวลาก่อนหน้า ทุกๆเซลมีกฎที่กำหนดเพื่อใช้ในการปรับปรุงค่าของเซล โดยการเปลี่ยนแปลงขึ้นอยู่กับเซลรอบข้าง ทุกๆเซลที่มีการปรับปรุงจะนำกฎที่มีไปใช้เพื่อสร้างสถานะใหม่เกิดขึ้น

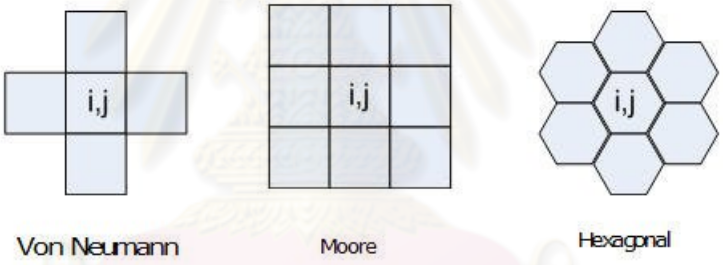
ตัวอย่างหนึ่งที่ใช้จำลองเซลลูลาร์ออโตมาแบบ 2 มิติ ได้แก่ การให้แต่ละเซลมี 2 สถานะที่เป็นไปได้คือ สีขาวกับสีดำ เซลรอบข้างมี 8 สี่เหลี่ยมที่สัมผัสกับสี่เหลี่ยมที่อยู่ตรงกลาง ดังนั้นเซลแต่ละเซลนั้นจะมีสถานะรอบตัวที่เป็นไปได้อยู่ทั้งหมด $2^9 = 512$ แบบ ซึ่งแต่ละแบบก็จะมีผลต่อการเปลี่ยนแปลงสถานะของเซลตรงกลางต่างๆกันไป ในเวลาต่อมา Conway's Game of Life ได้ใช้วิธีการนี้และวิธีการนี้ได้รับความนิยมยังแพร่หลาย โดยทั่วไปจะสมมติให้แต่ละเซลในตอนเริ่มต้นมีเพียงสถานะเดียว [3]

สำหรับเซลลูลาร์ออโตมาตาแบบ 1 มิตินั้น กำหนดให้เซลรอบข้างมีเซลข้างเคียงเพียงซ้ายกับขวาและสามารถกำหนดกฎได้ แม้ว่าเซลรอบข้างอาจใช้การเชื่อมกันแบบสมมาตรดังรูปที่ 2 ส่วนเซลลูลาร์ออโตมาตาแบบ 2 มิตินั้น เราสามารถกำหนดเซลลูลาร์ออโตมาตาในรูปทรงหลายเหลี่ยม เช่น สี่เหลี่ยมมี 4 เซลรอบข้าง หกเหลี่ยมมี 6 เซลรอบข้างดังรูปที่ 3 แต่ส่วนมากนิยมศึกษาเซลที่มีโครงสร้างเป็นรูปทรงสี่เหลี่ยมที่มีเซลรอบข้างเป็น 8 เซลซึ่งเป็นแบบที่เรียกว่า Moore neighborhood ดังกลางรูปที่ 3

รูปที่ 2 แสดงตัวอย่างของเซลลูลาร์ออโตมาตาแบบ 1 มิติโดยกฎของเซลนั้นใช้การกำหนด Rule 30 ซึ่งเริ่มจากการกำหนด 1 ไว้ตรงกลางของแต่ละรูป โดยที่สีขาวคือ ปิด สีดำคือ เปิด และกำหนดกฎของการเปลี่ยนแปลงดังรูปที่ 2



รูปที่ 2 แสดงเซลลูลาร์ออโตมาตาแบบ 1 มิติ ของ Rule 30 ที่มีการประมวลผล 15 ครั้ง [1]



รูปที่ 3 แสดงรูปแบบของเซลลูลาร์ออโตมาตาแบบ 2 มิติพร้อมเซลรอบข้าง [4]

2.1.2 อัลกอริทึมฟลอกกิง

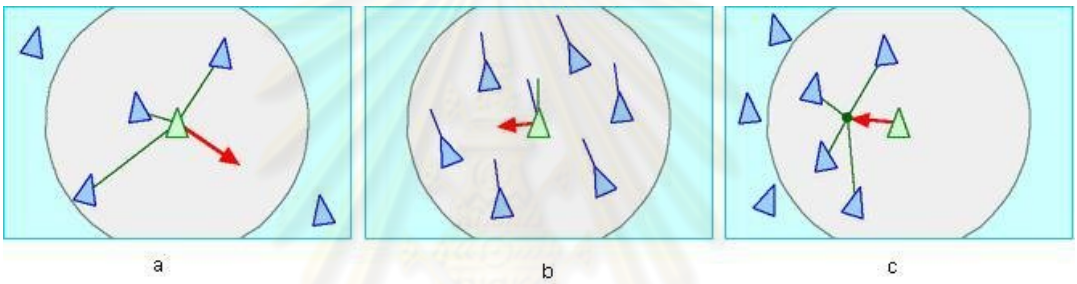
อัลกอริทึมฟลอกกิง ซึ่งถูกนำเสนอโดยเรโนลด์ [5] เป็นการจำลองพฤติกรรมของเอเจนต์เพื่อให้เอเจนต์เคลื่อนที่เป็นฝูงได้อย่างเป็นธรรมชาติ ตัวอย่างเช่น การบินของฝูงนก การว่ายน้ำของฝูงปลา และการเคลื่อนที่ของฝูงสัตว์บนพื้น

กฎของการควบคุมฝูงเอเจนต์ด้วยอัลกอริทึมนี้มี 3 ข้อ

1. การหลีกเลี่ยงการเบียดเสียดจากตัวข้างเคียง ทำโดยกำหนดระยะทางรอบแต่ละเอเจนต์ไว้ เมื่อหาเอเจนต์ตัวอื่นๆ ที่อยู่ภายในระยะนั้นได้แล้วจากนั้นนำมาคำนวณหาระยะทางเฉลี่ยระหว่างกันแล้วนำมาปรับก็ตำแหน่งที่เอเจนต์แต่ละตัวอยู่ ดังแสดงในรูปที่ 4 (a)

2. การเคลื่อนที่ไปในทิศทางใกล้เคียงกับตัวที่อยู่ข้างเคียง ทำโดยให้เอเจนต์แต่ละตัวทำการค้นหาตำแหน่งของตัวที่อยู่รอบข้างภายในขอบเขตที่ตนรับรู้ได้ก่อน จากนั้นทำการหาค่าเฉลี่ยของทิศทางจากเอเจนต์ที่อยู่รอบข้างและปรับทิศทางให้เข้ากับทิศทางนั้น ส่วนความเร็วนั้นจะต้องใกล้เคียงกับตัวที่อยู่ใกล้ ถ้าเกิดตัวที่นำอยู่มีความเร็วเพิ่มขึ้นสมาชิกแต่ละตัวก็จะมีความเร็วเพิ่มขึ้นตามไปด้วย ซึ่งแสดงในรูปที่ 4(b)

3. การรักษาระยะห่างจากตัวรอบข้าง โดยพยายามให้เอเจนต์เคลื่อนที่ไปยังตำแหน่งเฉลี่ยของตัวที่อยู่รอบข้าง เพื่อไม่ให้เอเจนต์เข้าใกล้กันมากจนเกินไปหรือออกห่างจากกลุ่มจนเกินไป ซึ่งแสดงในรูปที่ 4 (c)



รูปที่ 4 แสดงกฎของการควบคุมฝูงเอเจนต์ด้วยอัลกอริทึมฟลอกกิง [5]

ซึ่งกฎ 3 ข้อนี้สามารถทำให้การเคลื่อนที่ของฝูงสัตว์มีพฤติกรรมเป็นธรรมชาติ วิธีการนี้ได้ถูกนำไปใช้ในการทำสกรีนเซฟเวอร์และอนิเมชัน ซึ่งใช้ในการสร้างฝูงชนที่ต้องการให้เคลื่อนไหวแบบสมจริง เช่น ในภาพยนตร์เรื่องแบทแมน รีเทิร์น [6] (Batman Returns) ในปี 1992 ซึ่งกำหนดให้เป็นคุณสมบัติของฝูงเพนกวินที่เดินอยู่ในเมืองกอดแฮม และฝูงค้างคาว และในภาพยนตร์เรื่องไลออนคิง [7] (Lion King) ในปี 1994 ได้นำไปใช้ในส่วนของการวิ่งอย่างอลหม่านของฝูงวิลเดอบีส ดังแสดงในรูปที่ 5

สิ่งกีดขวางที่อยู่กับที่สามารถหลบได้โดยการกำหนดเส้นทางให้กับฝูงตั้งแต่เริ่มต้น หรือให้เอเจนต์ตรวจจับเอาเองโดยวิธีการให้สิ่งกีดขวางส่งสัญญาณเตือนว่ามีสิ่งกีดขวางอยู่แถวนั้นและให้เอเจนต์เปลี่ยนเส้นทางไปเอง

รูปที่ 5 แสดงการวิ่งอย่างอลหม่านของฝูงวิลเดอบีสในภาพยนตร์เรื่องไลออนคิง[7]

การหลบหลีกสิ่งกีดขวางในสิ่งแวดล้อมของฟลอกกิงอัลกอริทึมมี 2 วิธี [8]

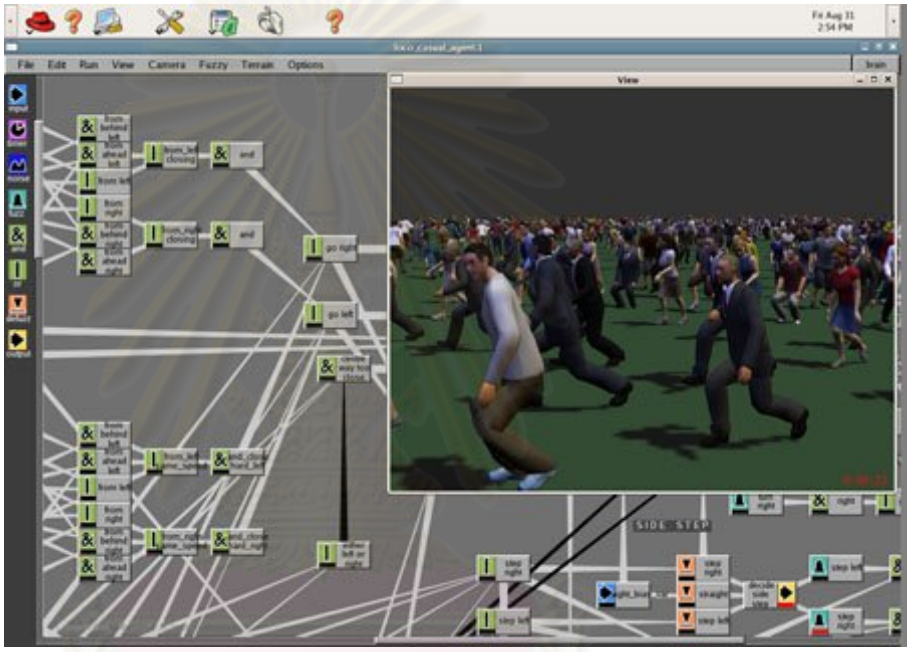
1. ใช้หลักการฟอร์สฟิลด์ (Force field) คือ การกำหนดให้สิ่งกีดขวางส่งสัญญาณที่เอเจนต์รับรู้ได้ออกมา เมื่อเอเจนต์เข้ามาใกล้ก็จะได้รับสัญญาณ ยิ่งเอเจนต์เคลื่อนที่เข้ามาใกล้มากเท่าไรความแรงของสัญญาณก็จะแรงขึ้น วิธีนี้ให้ผลลัพธ์ที่ดีแต่ก็มีบางกรณีต้องใช้เวลานานในการคำนวณมากกว่าปกติ

2. ใช้หลักการสเตียร์ ทู อะวอย (Steer to avoid) คือ การให้เอเจนต์มองเห็นระยะไกล เมื่อเอเจนต์มองเห็นสิ่งกีดขวางก็จะพยายามไม่เข้าไปใกล้โดยมีการประมาณความกว้างของสิ่งกีดขวางหรือพยายามเดินทางรอบสิ่งกีดขวางเพื่อมุ่งไปยังเป้าหมาย

2.1.3 มาสซีพ (massive)

มาสซีพ [9] คือ ชื่อของซอฟแวร์ที่ใช้ในการสร้างการจำลองพฤติกรรมของฝูงชนให้เกิดความสมจริง โดยนำเอเจนต์มาจำลองการเคลื่อนที่ของฝูงชนในหลายๆ ด้าน เช่น ใช้ในทางภาพยนตร์ เกมส์ โทรทัศน์ สถาปัตยกรรม การขนส่ง วิศวกรรม และ หุ่นยนตร์ เป็นต้น ผู้ใช้ซอฟแวร์

สามารถกำหนดให้เอเจนต์แต่ละตัวมีการท่าทางการแสดงออก และกำหนดรูปแบบการตอบสนองของเอเจนต์ได้ เอเจนต์ของมาสซีสมีสมองเป็นของตัวเอง ซึ่งผู้ใช้ซอฟต์แวร์ควบคุมการสั่งการของสมองของเอเจนต์โดยใช้อินเตอร์เฟสของฟัสซีโลจิก (Fuzzy logic) ซึ่งรูปแบบของคำสั่งจะครอบคลุมการโต้ตอบกับสิ่งแวดล้อมรวมไปถึงการแสดงออกทางอารมณ์ด้วย ตัวอย่างการนิยามคำสั่งด้วยฟัสซีโลจิกอินเตอร์เฟสสำหรับคนแต่ละคนในฝูงชนแสดงดังรูปที่ 6 โดยอินเตอร์เฟสจะมีลักษณะของการกำหนดทางเลือกและความสัมพันธ์ให้การตัดสินใจต่างๆ



รูปที่ 6 แสดงหน้าต่างการทำงานของมาสซีสซอฟแวร์ [9]

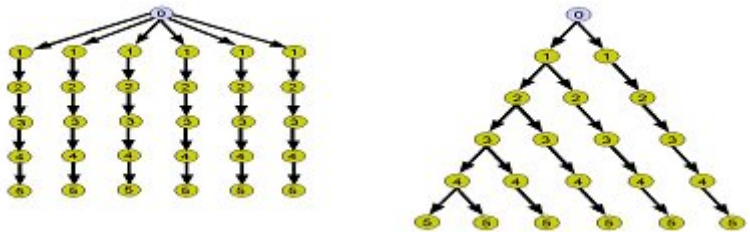
2.2 งานวิจัยที่เกี่ยวข้อง

Tantisiriwat [10] (A Crowd Simulation Using Individual-Knowledge-Merge base Path Construction and Smoothed Particle Hydrodynamics) ได้นำเสนอเทคนิคใหม่ที่ใช้ในการจำลองฝูงชนโดยใช้การหาเส้นทางแบบอัตโนมัติ ซึ่งเอเจนต์แต่ละตัวสามารถหาเส้นทางไปยังจุดหมายปลายทางได้เองโดยการดูจากแผนที่ที่มันสร้างขึ้น ซึ่งได้จากการมองเห็นและการสื่อสารระหว่างเอเจนต์ โดยที่เอเจนต์มีการรับรู้ 2 แบบ คือ การมองเห็น และการสื่อสาร ซึ่งในขณะที่เอเจนต์เคลื่อนที่เอเจนต์ก็จะทำการเรียนรู้สิ่งแวดแวดล้อมที่ได้เดินไปแล้วนำไปสร้างแผนที่ จากนั้นก็จะสร้างเส้นทางที่ให้เอเจนต์เดินโดยเส้นทางนั้นเอเจนต์จะไม่มีการชนกับสิ่งกีดขวางเลย แผนที่ที่ได้นี้สามารถเพิ่มเติมได้ตลอดเวลา เมื่อเอเจนต์มีการสื่อสารระหว่างกันจะมีการส่งข้อมูลระหว่างกัน เอเจนต์จะได้รับแผนที่ของเอเจนต์อีกตัวหนึ่งแล้วนำมาเพิ่มไปในแผนที่ของตัวเอง การ

เคลื่อนไหวของฝูงชนนั้นใช้หลักการการไหลของน้ำ เทคนิคการจำลองนี้จะไม่มีการคำนวณก่อนหน้าเลย การคำนวณจะทำเมื่อเอเจนต์ต้องการหาเส้นทางเท่านั้น แต่เทคนิคนี้ไม่สามารถทำงานได้ถูกต้องในครั้งเดียวกล่าวคือต้องมีการเก็บข้อมูลก่อน เอเจนต์แต่ละตัวต้องทำการเก็บข้อมูลและสร้างเส้นทางเอง งานวิทยานิพนธ์นี้แตกต่างกับงานวิจัยชิ้นนี้ตรงที่ไม่เน้นการหาเส้นทาง ซึ่งเส้นทางของเอเจนต์ในงานวิทยานิพนธ์นี้ได้จากการคำนวณอินฟลูเอนซ์แมปจากเซลของแผนที่โดยตรง นอกจากนี้งานวิทยานิพนธ์นี้ยังไม่ใช้สมการการคำนวณที่ซับซ้อนแบบงานวิจัยนี้ เนื่องจากมีความต้องการให้ผู้ใช้งานโปรแกรมต้นแบบซึ่งไม่มีความรู้ทางด้านสมการสามารถปรับเปลี่ยนพารามิเตอร์ของฝูงชนได้โดยง่าย

Saber [11] (Flocking with Obstacle Avoidance with Limited Communication in Mobile Networks) งานวิจัยนี้ใช้กราฟแทนฝูงเอเจนต์โดยใช้จุดแทนเอเจนต์และใช้เส้นในกราฟแทนระยะที่ประชิดกัน โดยการเคลื่อนของฝูงจะอยู่ในรูปของกราฟ และการหลบหลีกสิ่งกีดขวางจะทำในรูปแบบการประชิดของจุดกับเส้นแล้วให้จุดที่อยู่ประชิดสิ่งกีดขวางผลักดันให้จุดที่อยู่ประชิดกับมันอีกด้านหนึ่งเคลื่อนที่ไปโดยไม่ไปติดกับสิ่งกีดขวาง ซึ่งมีการแทนสิ่งกีดขวางเป็นจุดหรือเซตของจุด

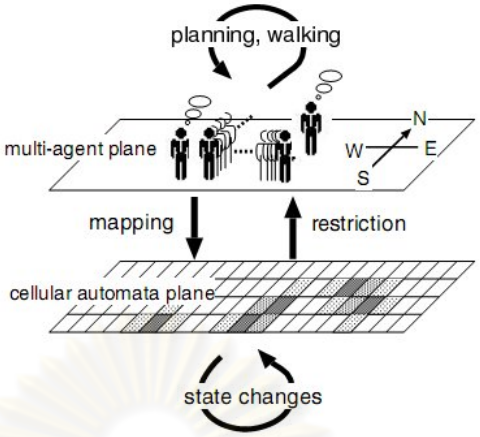
Leo [12] (Coherent Formation for Agents Using Flocking With Cellular Automata.) งานวิจัยชิ้นนี้เป็นการแสดงวิธีใช้งานฟลอกกิงร่วมกับเซลลูลาร์ออโตมาตา ซึ่งได้ใช้ 1 เซลต่อ 1 เอเจนต์โดยได้ทดลองกับระบบออฟไลน์และระบบออนไลน์ ซึ่งในระบบออนไลน์ได้ทดลองกับ เกมลินเนจ ทู การเคลื่อนที่ของเอเจนต์นั้นใช้แนวทางผู้นำผู้ตามโดยมีการตั้งกฎเกณฑ์ในการเลือกผู้นำ โดยที่การรวมกลุ่มของเอเจนต์เพื่อตามผู้นำนั้นจะอยู่ในรูปสี่เหลี่ยมและสามเหลี่ยมโดยจะมีรูปคล้ายกราฟต้นไม้ ดังรูปที่ 7 ซึ่งจะมีการรวมกลุ่มใหม่เมื่อพบสิ่งกีดขวาง โดยการจัดกลุ่มใหม่นั้นจะมีการใช้เวลาในการจัดกลุ่มแตกต่างกันกล่าวคือ ถ้าการเคลื่อนที่นั้นเป็นแบบทางเดิมไม่เปลี่ยนทิศทางการจัดกลุ่มใหม่ก็จะใช้เวลาน้อยกว่าเมื่อการเคลื่อนที่นั้นเป็นการเปลี่ยนทิศทาง การจัดรูปแบบการเรียงตัวของเอเจนต์นั้นทำโดยคำนวณที่อยู่ของเอเจนต์แต่ละตัวจากเซลลูลาร์ออโตมาตา เป็นการคำนวณโดยใช้หนึ่งเซลต่อหนึ่งเอเจนต์ งานวิทยานิพนธ์นี้ทำการประหยัดการคำนวณโดยให้หนึ่งเซลมีได้หลายเอเจนต์ ดังนั้นวิธีการจากงานวิจัยนี้จึงไม่สามารถนำมาใช้ได้เพราะจะเป็นการเพิ่มจำนวนเซล ซึ่งขัดกับความต้องการของงานวิทยานิพนธ์ที่จะลดการคำนวณ



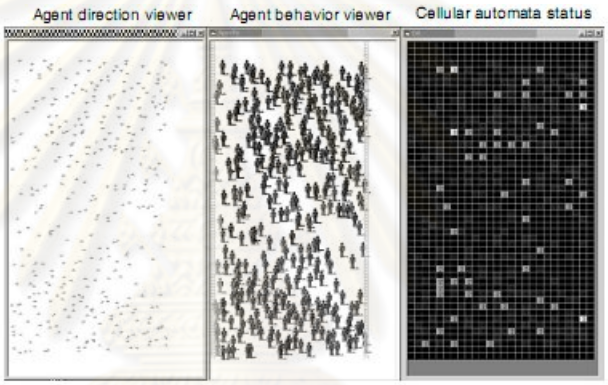
รูปที่ 7 แสดงการรวมกลุ่มรูปสี่เหลี่ยมและสามเหลี่ยม [12]

Masaru [13] (Massive Multi-Agent Simulation in 3D) งานวิจัยนี้ได้นำเสนอวิธีการจำลองฝูงชนในพื้นที่จำกัด ให้การเคลื่อนที่เหมือนธรรมชาติซึ่งเป็นเวลาจริง (real time) โดยการนำเอาอัลกอริทึมฟลอกกิงมาใช้โดยมีการปรับปรุงกฎของอัลกอริทึมข้อการหลีกเลี่ยงการเบียดเสียดจากตัวข้างเคียง โดยมีการเพิ่มกฎการหลบหลีกสิ่งกีดขวางและเพิ่มตัวแปรที่ใช้ในการแสดงพฤติกรรมที่แตกต่างกันเข้าไป อย่างไรก็ตาม เอเจนต์แต่ละตัวยังคงแยกกันคำนวณทำให้การประมวลผลยังคงมีข้อจำกัดอยู่

Hamagami [14] (Method of crowd simulation by using multi-agent on cellular automata) ได้นำเสนอการจำลองพฤติกรรมของฝูงชนในรูปแบบใหม่ เป็นแบบ 2 ชั้น ซึ่งประกอบด้วย ชั้นบนเป็นส่วนของ มัลติเอเจนต์ ชั้นล่างเป็นส่วนของเซลลูลาร์ออโตมาตาดังรูปที่ 8 ซึ่งทั้ง 2 ชั้นนี้มีความสัมพันธ์กันกล่าวคือชั้นบนเป็นส่วนแสดงพฤติกรรรมของฝูงชน ชั้นล่างเป็นตัวประมวลผลโดยทั้ง 2 ชั้นนี้ทำงานแบบคู่ขนาน และวิธีนี้ได้ใช้เอเจนต์ 2 แบบในการทดลอง คือ โฮโมจีเนียสเอเจนต์(เอเจนต์ที่ลักษณะเหมือนกัน) กับเฮทเทอรโรจีเนียสเอเจนต์(เอเจนต์ที่มีลักษณะที่แตกต่างกัน) มีการกำหนดคุณสมบัติให้กับเอเจนต์ 3 อย่างคือ ขอบเขตการมองเห็น ความเร็วในการเดิน และร่างกาย โดยทำการทดลองเปรียบกันระหว่างโฮโมจีเนียสเอเจนต์กับเฮทเทอรโรจีเนียสเอเจนต์ ซึ่งมีการกำหนดค่า ความหลากหลายของเอเจนต์ ค่าสิ่งแวดล้อม และระยะในการมองเห็นที่แตกต่างกัน จากการทดลองปรากฏว่าเฮทเทอรโรจีเนียสเอเจนต์สามารถเดินได้เหมือนคนทั่วไปคือการเดินไม่มีการชนซึ่งแตกต่างโฮโมจีเนียสที่เมื่อเกิดการชนกันจะเดินวนเป็นวงกลมและทำให้เกิดช่องว่างดังแสดงในรูปที่ 9 การจำลองนี้ได้ใช้เซลลูลาร์ออโตมาตา 1 เซลต่อการจำลองคน 1 คน การทำงานของการจำลองพฤติกรรมของฝูงชนในรูปแบบนี้ทำงานได้เฉพาะในระนาบของพื้นดิน การทำงานไม่ได้อาศัยการตัดสินใจเป็นกลุ่มอย่างงานวิทยานิพนธ์นี้



รูปที่ 8 แสดงความสัมพันธ์ของเลเยอร์ทั้ง 2 ชั้น [14]

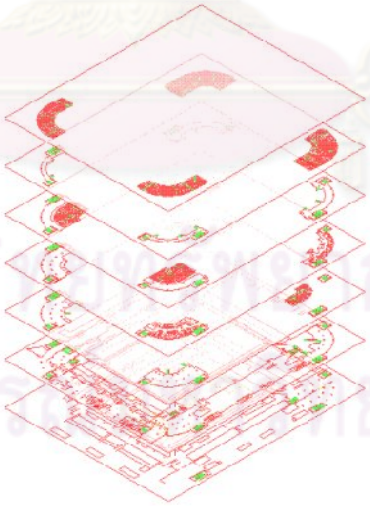


รูปที่ 9 แสดงหน้าจอแสดงผลการเดินของเอเจนต์โดยเซลลูลาร์ออโตมาตา [14]

Bandini [15] (Situated Cellular Agents for Crowd Simulation and Visualization) ได้นำเสนอการใช้ระบบมัลติเอเจนต์ในการทำแบบจำลองฝูงชนโดยใช้หลักการพื้นฐานมาจาก Situated Cellular Agents(SCA) ซึ่ง Situated Cellular Agents เป็นคลาสพิเศษของ Multilayer Multi Agent Situated system ซึ่งมีพื้นฐานมาจาก เซลลูลาร์ออโตมาตา เอเจนต์ที่ใช้ในระบบจำลองนี้เป็นแบบอัตโนมัติมีพฤติกรรมเอกลักษณ์ส่วนตัว และครอบครองพื้นที่ไม่เท่ากัน เอเจนต์ต่างชนิดกันก็มีเอกลักษณ์ที่ต่างกัน แบบจำลอง SCA ได้กำหนด multi agent system เป็นสถานการณ์ในสิ่งแวดล้อม เอเจนต์มีการโต้ตอบ 2 แบบ คือ ถ้าเอเจนต์อยู่ในสิ่งแวดล้อมมันสามารถเปลี่ยนสถานะและโต้ตอบโดยการแพร่ผ่านช่องว่าง ค่าการแพร่คิดการแพร่ผ่านที่ว่างตามฟังก์ชั่นการแพร่ กระบวนการนี้อธิบายลักษณะของเอเจนต์โดยกำหนดการกระทำที่เป็นไปได้และกระบวนการเลือกการโต้ตอบภายใต้สถานะภายในกับตำแหน่งของเอเจนต์ การกระทำที่เป็นไปได้สามารถเปลี่ยนแปลงได้เมื่อเอเจนต์เปลี่ยนตำแหน่ง การกระทำที่เป็นไปได้มี 2

แบบคือ 1. เปลี่ยนสถานะหรือตำแหน่ง 2. โต้ตอบกับเอเจนต์ตัวอื่น การรับรู้ของเอเจนต์ขึ้นอยู่กับตำแหน่งที่อยู่ในสิ่งแวดล้อมและการเลือกการโต้ตอบจากการกระทำที่เป็นไปได้ สถานะของเอเจนต์ก็มีส่วนต่อการรับรู้ของเอเจนต์ งานวิจัยนี้ได้ใช้เซลลูลาร์ออโตมาตาในการคำนวณและใช้แมกซ์สคริปต์ ในการสร้างสิ่งแวดล้อมโดยให้ทั้งสองอย่างทำงานประสานกันโดยในส่วนของการสั่งงานให้เอเจนต์เคลื่อนที่เป็นส่วนของเซลลูลาร์ออโตมาตา แต่ยังคงใช้งาน 1 เอเจนต์ต่อ 1 เซล และเอเจนต์ได้รับการนำทางจากวัตถุด้วยโดยวัตถุทำหน้าที่ส่งสัญญาณไปยังเอเจนต์

Klupfel [16] (Simulation of the Evacuation of a football stadium using the CA Model PedGO) ได้นำเสนอโปรแกรมชื่อ PedGo ซึ่งใช้เซลลูลาร์ออโตมาตาแบบ 2 มิติ และแบบจำลองการอพยพจากสถานที่กว้างใหญ่และซับซ้อน ซึ่งได้ใช้การสร้างสถานการณ์สมมติคือการอพยพคนออกจากสนามฟุตบอลสูง 7 ชั้นโดยให้อพยพผ่านทางบันไดเท่านั้น โดยใช้เซลลูลาร์ออโตมาตา ในการประมวลผลหาความน่าจะเป็นของการเคลื่อนที่ การปรับตัวของเอเจนต์ที่ออกไปนอกสนาม ข้อมูลในเซลจะมีระยะทางที่ไปถึงทางออกต่อไป การอพยพทำโดยการแบ่งกลุ่มคนโดยที่แต่ละกลุ่มมีความสามารถต่างกัน โดยใช้ 1 เซลต่อ 1 เอเจนต์ ซึ่งเอเจนต์ไม่มีการรับรู้ข้อมูลของกันและกัน รูปที่ 10 แสดงผังของสนามฟุตบอลที่ใช้ในการกำหนดพื้นที่เซล

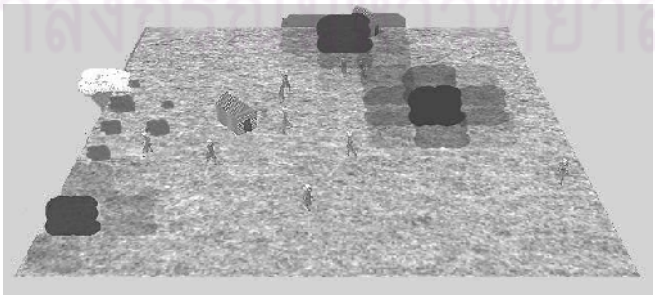


รูปที่ 10 แสดงโครงสร้างของสนามฟุตบอล [16]

จะเห็นว่า งานจำลองฝูงชนด้วยเซลลูลาร์ออโตมาตานั้น ส่วนมากเน้นไปที่การจำลองในระดับหนึ่งเซลต่อหนึ่งเอเจนต์ งานวิทยานิพนธ์นี้ต้องการนำการจำลองฝูงชนไปใช้โดยลด

การคำนวณของแต่ละเอเจนต์ลง เพื่อใช้กับการจำลองเอเจนต์จำนวนมากขึ้น จึงเปลี่ยนจากการให้แต่ละเอเจนต์ตัดสินใจด้วยตัวเอง เป็นให้เซลตัดสินใจแทนเอเจนต์ที่อยู่ข้างในนั้นให้มากที่สุด

สำหรับงานวิจัยที่หนึ่งเซลสามารถใช้เก็บเอเจนต์ได้หลายตัวนั้น จะไม่ใช่การจำลองฝูงชน Sweetser [17] (Combining Influence Maps and Cellular Automata for Reactive Game Agent) นำเสนอแบบจำลองการออกแบบเอเจนต์ที่ได้พัฒนารวมกับเซลลูลาร์ออโตมาตาแบบหลายชั้น และมีแบบจำลองสิ่งแวดล้อมในแผนที่ที่สามารถชักจูงเอเจนต์ให้ตัดสินใจเคลื่อนที่ไปในทิศทางต่างๆได้ sweetser พัฒนาระบบอีเมอเจนต์ (EmerGEnt system) ซึ่งสามารถนำไปใช้กับเอเจนต์เกมที่สามารถตอบสนองกับปรากฏการณ์ทางธรรมชาติได้ ซึ่งระบบอีเมอร์เจนต์นี้อาศัยหลักการทางเซลลูลาร์ออโตมาตา โดยที่แต่ละเซลมีการเก็บค่า ความสูง ความดัน อุณหภูมิ การไหลของน้ำ พื้นดิน เมื่อเอเจนต์ได้เลือกจุดหมายปลายทางแล้วเอเจนต์จะเคลื่อนที่ไปยังจุดนั้น โดยระหว่างทางนั้น เอเจนต์ใช้ฟังก์ชั่นความสบายใจในการตัดสินใจที่จะเคลื่อนที่หรือหยุดนิ่ง กล่าวคือ เอเจนต์จะเคลื่อนที่จากที่ที่มีค่าความสบายน้อยไปยังที่ที่มีค่าความสบายมากกว่าโดยค่าความสบายนี้ได้มาจากการคำนวณฟังก์ชันความสบายซึ่งคำนวณจากผลรวมของค่าน้ำหนักคูณกับค่าที่ไฟไหม้อยู่ในแผนที่ ค่าความร้อน ค่าความดันและค่าความชื้น ซึ่งจะเลือกค่าที่น้อยที่สุด ดังนั้นค่าที่ออกมาจะอยู่ระหว่าง 0 ถึง 1 ถ้าค่าความสบายที่ได้รับมีค่าน้อยกว่า 0.1เอเจนต์จะไม่เคลื่อนที่ ถ้าค่าความสบายอยู่ระหว่าง 0.1-0.3 เอเจนต์จะเคลื่อนที่ไปยังเซลที่มีความสบายมากกว่า ถ้าค่าความสบายอยู่ระหว่าง 0.3-0.6 เอเจนต์จะวิ่งออกจากเซล แต่ถ้าค่าความสบายมากกว่า 0.6 เอเจนต์จะวิ่งด้วยความเร็วออกจากเซล Sweetser ได้ทำการทดลองปรับค่าความสำคัญที่เอเจนต์ให้กับความสบายและกับจุดหมาย ผลการทดลองปรากฏว่าสามารถสร้างเอเจนต์ที่หลบหลีกอันตรายเพื่อไปยังเป้าหมายได้ ตัวอย่างสถานการณ์การทดลองแสดงในรูปที่ 11

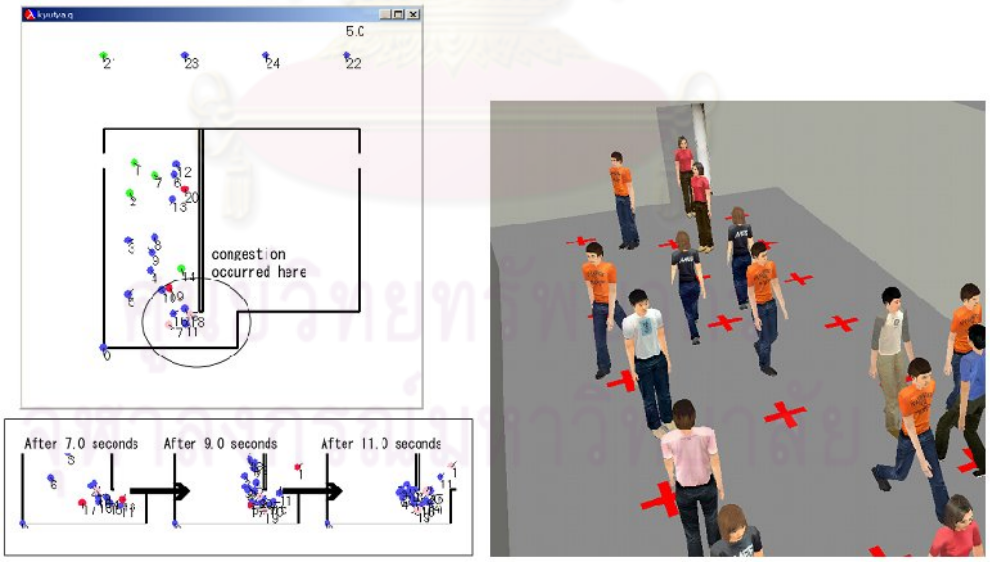


รูปที่ 11 แสดงสถานะเริ่มต้นที่ได้จากการสุ่มตำแหน่งที่ฝนตกและการระเบิด ก่อนเกิดไฟไหม้ [17]

งานของ Sweetser นี้ เซลลูลาร์ออโตมาตาหนึ่งเซลสามารถมีเอเจนต์ได้หลายตัว เอเจนต์แต่ละตัวอ่านค่าจากเซลที่ตนอยู่และอ่านค่าจากเซลรอบข้างตามระยะทางที่ตนมองเห็น แล้วตัดสินใจเคลื่อนไหว

งานวิทยานิพนธ์นี้ใช้เซลหนึ่งเซลเก็บเอเจนต์ได้หลายตัวเช่นเดียวกัน แต่เอเจนต์แต่ละตัวจะไม่อ่านค่าจากเซลรอบข้างแล้วตัดสินใจทีละตัว เพราะตัวเซลเองจะเป็นตัวอ่านค่าจากเซลรอบข้าง แล้วทำการตัดสินใจแทนเอเจนต์ทุกตัวที่อยู่ในเซลนั้นเพื่อลดการคำนวณในส่วนเอเจนต์

Murakami [18] (Multi-Agent Simulation for Crisis Management) ในงานวิจัยนี้ผู้วิจัยได้เอา ระบบผู้นำ-ผู้ตามไปใช้เพื่อทำให้ระบบจำลองสมจริงมากขึ้น ระบบของเอเจนต์ในสถานการณ์นี้ มี 2 แบบคือ Free Walk กับ Flat Walk ซึ่ง Flat Walk คือ เป็นระบบที่สามารถควบคุมเอเจนต์จำนวน 20 ตัวหรือมากกว่าได้ ส่วน Free Walk คือ เป็นระบบที่ควบคุมเอเจนต์ในรูปแบบ 3 มิติ ที่เอเจนต์สามารถคุยกันได้และมองเห็นกันได้



(a) (b)

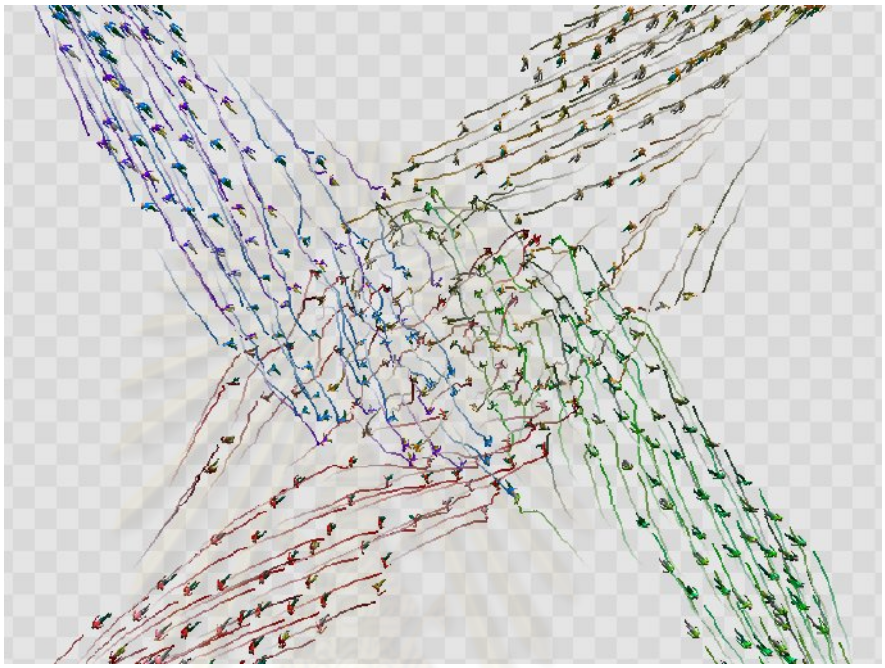
รูปที่ 12 แสดงแบบจำลอง 2 มิติของ Flat Walk รูป a และแบบจำลอง 3 มิติของ Free Walk ในรูป b [18]

ในการทดลองนี้ได้ทำการทดสอบวิธีการอพยพคน 2 วิธี คือ follow direction method (ผู้นำส่งเสียงบอกทางให้ผู้อพยพไปยังทางออก) กับ Follow me method (ผู้นำจะพาผู้อพยพออกไปยังทางออก) ดังรูปที่ 12 จากการทดลองปรากฏว่า การอพยพคนโดยใช้ Follow me method สามารถพาผู้อพยพทั้งหมดไปยังที่ที่ปลอดภัยได้อย่างรวดเร็วทั้งหมดถ้ามีจำนวนผู้นำมากเพียงพอ ส่วน Follow direction method นั้นก็สามารถพาผู้อพยพออกไปยังที่ที่ปลอดภัยได้ แต่จะช้าถ้ามีจำนวนผู้นำมาก งานนี้นำแนวคิดผู้นำ-ผู้ตามมาใช้งาน ทำให้เอเจนต์สามารถแสดงพฤติกรรมการเคลื่อนไหวเป็นกลุ่มออกมาได้เหมือนกับการทดลองโดยใช้คนจริง ในงานวิทยานิพนธ์จะนำความสัมพันธ์ของเอเจนต์ที่อยู่รอบข้างมาใช้งานเช่นเดียวกันทั้งระดับต่ำ (ทำฟลอกกิงอัลกอริทึมกับเอเจนต์ที่อยู่ใกล้กัน) และระดับสูง(ข้อมูลการเคลื่อนที่ของเอเจนต์ในเซลจะถูกส่งไปให้เซลรอบข้าง)

Treuille [19] (Continuum Crowds) ได้นำเสนอแบบจำลองฝูงชนแบบทันกาลโดยใช้หลักการของความต่อเนื่องแบบพลวัต ซึ่งได้พยายามลดการทำงานของเอเจนต์ โดยพยายามให้เอเจนต์คำนวณน้อยที่สุด ในการกำกับเส้นทางการเดินนั้นใช้ไดนามิคโพเทนเชียลฟังก์ชันกำหนดการเคลื่อนไหวของทุกตัวละครพร้อมกัน โดยไม่จำเป็นต้องใช้การตรวจการชนกันเลย ซึ่งการคำนวณส่วนใหญ่เป็นหน้าที่ของเซลและจะคำนวณแบบทีเดียวหมดยกเว้นการคำนวณความเร็วในการเดินของเอเจนต์ เซลทำการคำนวณและเลือกเส้นทางที่ดีที่สุดให้โดยที่เส้นทางเส้นนั้นจะไม่เกิดการชนกันในระหว่างที่เอเจนต์เดินเลย ตัวอย่างการเดินตัดกันของคนสี่กลุ่มที่ใช้วิธีการนี้แสดงในรูปที่ 13 การคำนวณต่างๆ อาศัยสมการทางคณิตศาสตร์ การคำนวณความเร็วจะใช้กริดโดยอาศัยค่าความหนาแน่นของเอเจนต์ที่อยู่รอบข้างนั้น แต่การที่ทำการคำนวณในระดับเซลโดยไม่คำนวณที่ระดับเอเจนต์เลยนั้น ทำให้ตัวละครที่อยู่ในเซลเดียวกันสามารถเดินชนกันได้ ผู้พัฒนาได้แก้จุดอ่อนนี้ด้วยการกำหนดระยะห่างที่น้อยที่สุดระหว่างตัวละครไว้ แต่ยังคงมีพฤติกรรมที่ไม่ควรเกิดแสดงออกมา เนื่องจากไม่ได้ทำการจำลองเอเจนต์แต่ละตัวเลย ลักษณะที่เอเจนต์แสดงออกมาจะเป็นในลักษณะเดียวกันมีความแตกต่างกันน้อยทำให้ใช้กับการจำลองฝูงชนได้ในระยะไกลเท่านั้น การนำระบบไปใช้งานทำได้ยากเนื่องจากผู้ใช้จำต้องปรับแต่งสมการด้วยตนเอง

ในงานวิทยานิพนธ์นี้จะใช้เซลในการคำนวณเช่นเดียวกัน ดังนั้นจะมีจุดด้อยเช่นกันคือตัวละครยังสามารถเดินชนกันได้และเอเจนต์มีลักษณะการเดินคล้ายคลึงกัน แต่วิทยานิพนธ์นี้จะไม่ใช้สมการที่ซับซ้อนในการคำนวณทิศทาง จะใช้สมการความสบายในการ

หลีกเลี่ยงการชนระหว่างเซลและการหาทิศทางเฉลี่ยของเอเจนต์เข้ามาแทนเพื่อให้ระบบสามารถใช้งานและปรับแต่งโดยผู้ไม่เชี่ยวชาญได้

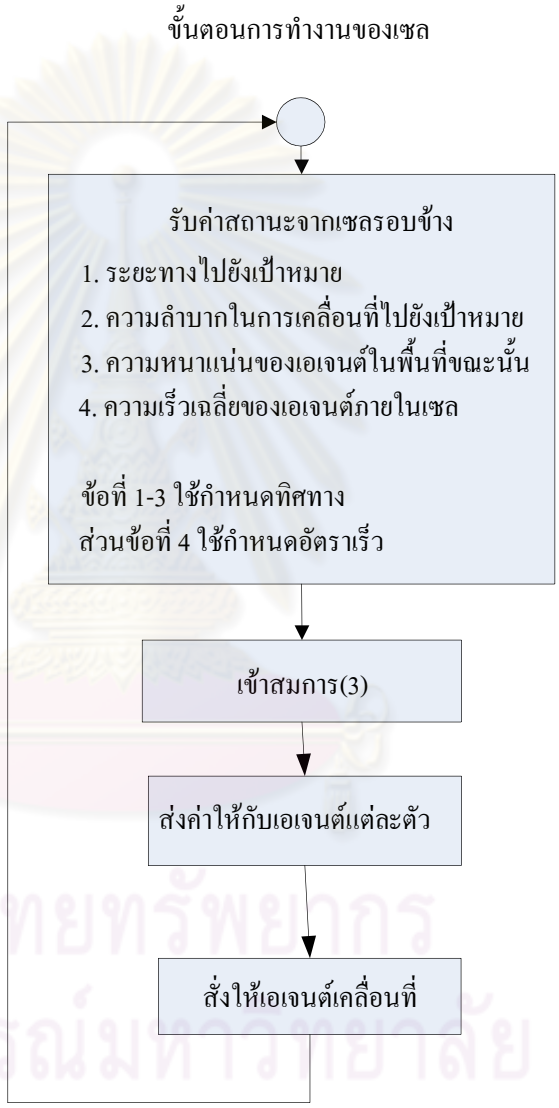


รูปที่ 13 แสดงการเดินตัดกันของฝูงคน 4 กลุ่ม [19]

บทที่ 3

ขั้นตอนการทำงานของโปรแกรม

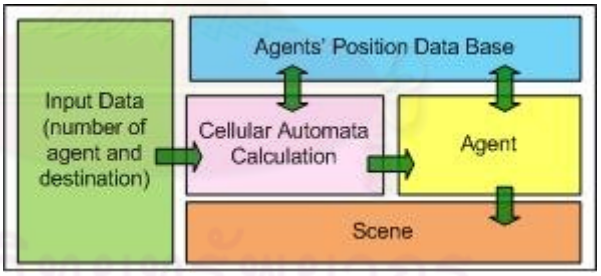
3.1 การทำงานโดยรวมของโปรแกรม



รูปที่ 14 แสดงการทำงานโดยรวมของระบบ

รูปที่ 14 แสดงการทำงานโดยรวมของโปรแกรม หลักการทำงานโดยรวมของระบบคือ ในทุกๆช่วงเวลาที่กำหนด (สำหรับระบบที่สร้างขึ้นนี้ มีช่วงเวลาที่กำหนดเท่ากับ 10 เฟรม) แต่ละเซลจะรับข้อมูลจากเซลรอบข้าง แล้วคำนวณทิศทาง และอัตราเร็วโดยเฉลี่ยให้กับเอเจนต์ทุกตัวในเซลนั้น จากนั้นแต่ละเซลจึงสั่งการให้เอเจนต์ภายในเซลเคลื่อนที่ตามที่คำนวณไว้

เริ่มต้นนั้น โปรแกรมจะรับข้อมูลที่ผู้ใช้กรอกเข้ามา สิ่งที่รับเข้ามาประกอบด้วย จำนวนเอเจนต์ ลักษณะการเรียงตัวของเอเจนต์ภายในแต่ละเซล โดยสภาพแวดล้อมของระบบที่ทดลองนั้น มีเซลทั้งหมด 100 เซล โดยแต่ละเซลมีจำนวนเอเจนต์ได้สูงสุด 16 เอเจนต์ และนอกจากนี้ยังมีข้อมูลจุดหมายปลายทางที่ต้องการให้ไป เมื่อโปรแกรมเริ่มทำงาน ทิศทางและเป้าหมายของการเคลื่อนที่ของฝูงชนในแต่ละเซลนั้น จะถูกกำหนดเบื้องต้นอย่างอัตโนมัติด้วยค่าความใกล้จุดหมายของเซลนั้นๆ ซึ่งค่าความใกล้นี้สามารถตกทอดจากเซลเป้าหมายไปยังเซลรอบข้างในแผนที่ได้ การเคลื่อนที่ของเอเจนต์ในระยะเริ่มแรกนั้นจะเคลื่อนที่ไปยังเซลที่มีค่าใกล้จุดหมายมากที่สุดเป็นสำคัญ ถ้ามีจุดหมายหลายจุดหมายที่มีความใกล้เท่ากัน เซลจะเลือกให้เอเจนต์แต่ละตัวในเซลมีเป้าหมายสลับกันไปด้วยอัตราส่วนเท่ากัน เมื่อมีการเคลื่อนที่ของฝูงชนเกิดขึ้นแล้ว ลักษณะการเคลื่อนที่โดยรวมภายในแต่ละเซลจะถูกแปลงไปเป็นพารามิเตอร์ของเซลนั้นๆ เพื่อเสริมกับค่าความใกล้จุดหมายเพิ่มเติมจากตอนที่ยังไม่มีการเคลื่อนที่ การเคลื่อนที่ของเอเจนต์จะเคลื่อนที่ด้วยความเร็วตามที่เซลกำหนดมาแต่สามารถเปลี่ยนแปลงได้จากระยะห่างระหว่างตัวมันเองกับกลุ่มเอเจนต์ตัวอื่นๆและระยะทางจากจุดหมาย โดยการเปลี่ยนแปลงนั้นเกิดจากการคำนวณในระดับเซลตลอดการเคลื่อนที่ การคำนวณในระดับเอเจนต์จะไม่มีการกระทำ รูปที่ 15 แสดงส่วนต่างๆของระบบ



รูปที่ 15 แสดงส่วนประกอบโดยรวมของระบบ

3.2 การทำงานของเซลลูลาร์ออโตมาตา

ค่าความใกล้จุดหมายของแต่ละเซล จะมีหน่วยเป็นจำนวนช่องเซลที่ตัวละครต้องเคลื่อนที่ผ่านเพื่อไปจากเซลปัจจุบันนั้นจนถึงเซลจุดหมาย ในการอิมพลีเม้นท์นั้นใช้อาร์เรย์สองมิติ หลังจากได้ค่าความใกล้จุดหมาย เซลจะต้องคำนวณค่าความหนาแน่นของประชากรภายในเซล เมื่อได้ค่าทั้ง 2 นี้จะนำไปคำนวณค่า “ความสบาย” ของเซล ซึ่งค่าความสบายนี้คำนวณได้จากสมการที่ 1 ค่าความสบายนี้จะถูกนำไปใช้ในการเลือกทิศทางที่เอเจนต์ต้องเคลื่อนที่เพื่อให้กลุ่ม

ของเอเจนต์หลีกเลี่ยงการชนกับกลุ่มอื่น โดยเซลจะเลือกให้เอเจนต์เคลื่อนที่ไปยังเซลที่มีค่าน้อยที่สุดจากเซลที่อยู่รอบข้างทั้ง 8 ด้าน

$$Comfort_i = W_{den} * Density_i + W_{dis} * Dis\tan ce_i \qquad (1)$$

เมื่อกำหนดให้

$Comfort_i$ คือ ค่าความสบายของเซล i

$Density_i$ คือความหนาแน่นของประชากรในเซล i ซึ่งยิ่งมีค่าน้อยยิ่งถือว่าดี

$Dis\tan ce_i$ คือระยะทางจากจุดหมายของเซล i ซึ่งยิ่งมีค่าน้อยยิ่งถือว่าดี

W_{den} กับ W_{dis} คือค่าน้ำหนักที่เอาไปคูณเพื่อปรับความสำคัญของความหนาแน่นและระยะทาง การถ่วงน้ำหนักในสมการที่ 1 นั้นผู้วิจัยได้ทำการจับคู่ค่าน้ำหนักจำนวน 3 คู่ ได้แก่ W_{den} = 0.3 W_{dis} = 0.7 W_{den} = 0.2 W_{dis} = 0.8 และ W_{den} = 0.1 W_{dis} = 0.9 การที่ผู้วิจัยให้ค่าหนักกับทางด้านระยะทางมากกว่าเนื่องจาก ต้องการให้เอเจนต์เคลื่อนที่ไปในทิศทางที่ใกล้ที่สุดและตรงไปตามรูปแบบของสถานการณ์จำลองต้นแบบ ซึ่งค่าน้ำหนักทั้ง 3 คู่ที่กล่าวมานั้นสามารถใช้เป็นค่าน้ำหนักได้ทุกคู่ แต่จากผลการทดลองนั้น เอเจนต์มีพฤติกรรมที่ดีที่สุดเมื่อค่า W_{den} = 0.3 และ W_{dis} = 0.7

สำหรับการเลือกทิศทางการเคลื่อนไหวรอบเซลแต่ละเซลนั้น เลือกจากค่า Comfort ของเซลที่อยู่รอบข้าง โดยเซลที่ถูกเลือกต้องมีค่าที่น้อยสุด

ความหนาแน่นของประชากรในเซล (Density) คือ อัตราส่วนของจำนวนเอเจนต์ต่อขนาดพื้นที่ซึ่ง ค่าความหนาแน่นขึ้นกับขนาดของเอเจนต์แต่ละตัวและขนาดเอเจนต์ ค่าความหนาแน่นของประชากรในแต่ละเซลนั้นหาได้จาก อัตราส่วนของจำนวนของเอเจนต์ที่อยู่ในเซลกับจำนวนเอเจนต์มากที่สุดที่เซลสามารถรับได้ ดังแสดงในสมการที่ 2

$$Density = \frac{n}{m} \qquad (2)$$

เมื่อ n = จำนวนเอเจนต์ที่อยู่ในเซล

m = จำนวนเอเจนต์ที่มากที่สุดเท่าที่เซลรับได้

ระยะทาง (Distance) คือระยะทางที่วัดจากเซลนั้นๆ ไปถึงเซลเป้าหมาย สำหรับการคำนวณหาระยะทางของเซลหนึ่งจากเซลจุดหมายนั้น หาได้จากการใช้อินฟลูเอนซ์แมปหรือการแพร่ของค่าระยะทาง ดังรูปที่ 16 จากรูป ถ้าเซล T คือเซลเป้าหมาย โดยคิดระยะทางที่เอเจนต์ต้องเคลื่อนที่ยังเซลเป้าหมายจากระยะการกระจัดของพิกัดเซลนั้นๆ กับเซลเป้าหมายซึ่งค่าที่ได้จะอยู่รูปของหน่วยเซล ซึ่งก็คือจำนวนเซลที่เอเจนต์จะต้องเคลื่อนที่ผ่านไปจนถึงเซลเป้าหมาย

2.828	2.236	2.0	2.236	2.828
2.236	1.414	1.0	1.414	2.236
2.0	1.0	T	1.0	2.0
2.236	1.414	1.0	1.414	2.236
2.828	2.236	2.0	2.236	2.828

รูปที่ 16 แสดงค่าระยะทางจากจุดหมายที่คำนวณโดยใช้อินฟลูเอนซ์แมป

ในสถานะแรกนั้นการหาทิศทางของเซลนั้น ผู้จัดทำใช้วิธีการคำนวณ โดยอาศัยทฤษฎีบทปีทาโกรัส โดยอาศัยผลต่างของพิกัดของเซลเป้าหมายลบด้วยพิกัดของเซลปัจจุบัน ซึ่งผลต่างที่ได้จะนำไปยกกำลังสองแล้วนำมารวมกันดังแสดงในรูปที่ 17 จากนั้นใช้ความสัมพันธ์ของสามเหลี่ยมมุมฉากเข้ามาหาค่าของมุม ซึ่งทำโดยจะหามุมที่เซลปัจจุบันทำกับเซลเป้าหมายก่อน ในที่นี้เราคิดค่าของมุมจากค่าของฟังก์ชันไซน์ ซึ่งเป็นค่าที่ได้จากทฤษฎีบทปีทาโกรัสโดยใช้สมการที่ 3 ซึ่งจะได้ค่าของ $\sin\theta$ มาซึ่งจะต้องทำการหาค่า θ จากการทำอินเวอร์สไซน์ ดังสมการที่ 4 เนื่องจากนำฟังก์ชันไซน์มาใช้นั้นจะเกิดปัญหาในกรณีที่ $\Delta y = 0$ นั่นก็คือค่าเซต้าที่ได้จะผิดไปจากความเป็นจริง จึงใช้ฟังก์ชันโคไซน์เข้ามาช่วยแก้ปัญหาในกรณีนี้ ฟังก์ชันโคไซน์แสดงในสมการที่ 5 และหาค่าเซต้าจากสมการที่ 6 จากนั้นนำค่าที่ได้ไปหามุมที่แท้จริง โดยจะทำการเปรียบเทียบค่า Δx กับ Δy ที่หาได้จากการคำนวณในทฤษฎีบทปิทาโกรัสในขั้นต้น ซึ่งค่า Δx และ Δy คือผลต่างของระยะห่างในแนวแกนเอ๊กซ์และในแนวแกนวาย ตามลำดับ มาเทียบกับค่าของควอแดรนท์ (quadrant) จากนั้นนำค่าเซต้าที่ได้ไปทำการหาค่ามุมจากจตุภาคต่างๆ ซึ่งทำการเปรียบเทียบกับสมการที่ 7 ซึ่งค่าที่ได้มานี้จะเป็นค่าที่เซลปัจจุบันทำมุมกับเซลเป้าหมายนั่นก็คือมุมที่ใช้ในการเคลื่อนที่ไปยังเป้าหมายของเซลนั้น หลังจากนั้นเรานำค่ามุมที่ได้มาเทียบกับมุมในรูปที่ 18 เพื่อ

จำกัดการเปลี่ยนของมุมไม่ให้มากเกินไปจากมุมเดิมในตอนที่เข้าสมการที่ 8 โดยกำหนดค่ามุมที่เป็นไปได้มากที่สุดของทิศนั้นเพื่อใช้เป็นขอบเขตอ้างอิงทิศของมุมที่ใช้ในการเคลื่อนที่ของแต่ละเซล เมื่อเซลได้คำนวณมุมครบทุกเซลแล้วจะนำไปสู่ขั้นตอนการเปลี่ยนสถานะ

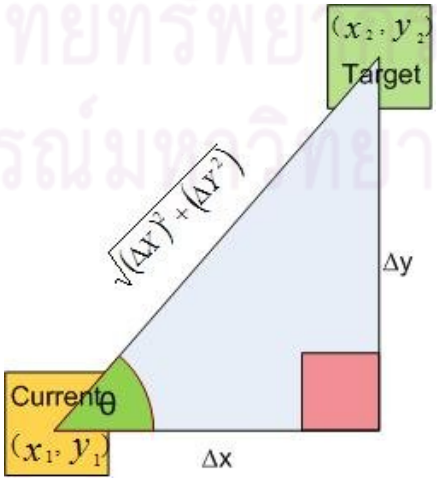
$$\sin\theta = \frac{\Delta y}{\sqrt{\left(\Delta x^2 + \Delta y^2\right)}} \qquad (3)$$

$$\theta = \arcsin\left(\frac{\Delta y}{\sqrt{\left(\Delta x^2 + \Delta y^2\right)}}\right) \qquad (4)$$

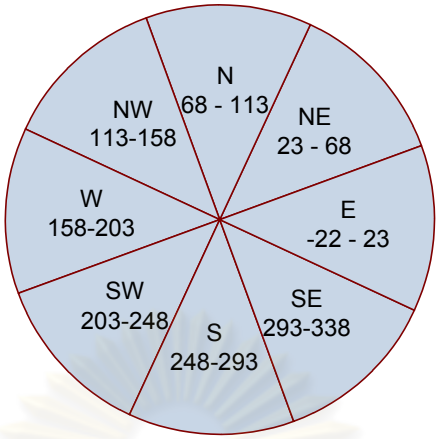
$$\cos\theta = \frac{\Delta x}{\sqrt{\left(\Delta x^2 + \Delta y^2\right)}} \qquad (5)$$

$$\theta = \arccos\left(\frac{\Delta x}{\sqrt{\left(\Delta x^2 + \Delta y^2\right)}}\right) \qquad (6)$$

$$\left.\begin{array}{llll} \Delta x > 0 & and & \Delta y > 0 & \rightarrow \quad \theta = \theta \\ \Delta x < 0 & and & \Delta y > 0 & \rightarrow \quad \theta = 180 - \theta \\ \Delta x < 0 & and & \Delta y < 0 & \rightarrow \quad \theta = 180 - \theta \\ \Delta x > 0 & and & \Delta y < 0 & \rightarrow \quad \theta = 360 - \theta \end{array}\right\} \qquad (7)$$



รูปที่ 17 แสดงวิธีหาค่ามุมของทฤษฎีปีทากอรัส



รูปที่ 18 แสดงทิศทางที่เป็นไปได้ จากการคำนวณในสถานะแรก

สำหรับการคำนวณทิศทางในสถานะอื่นๆนอกจากสถานะแรกนั้นค่าทิศทางโดยละเอียดนั้นหาจากค่าทิศทางเฉลี่ยโดยนำค่าทิศทางจากเซลรอบข้างทั้ง 8 มาถ่วงน้ำหนักกับทิศทางที่เซลนั้นเคลื่อนที่ไปยังเซลที่มีค่าความสบายน้อยที่สุด ดังแสดงในสมการที่ 8 รูปที่ 19(1) แสดงทิศทางที่ได้ของเซลที่มีสีเหลืองหมายเลข 8 ก่อนการใช้สมการ 8 โดยเซลที่มีชมพูหมายเลข 6 เป็นเซลที่มีค่าความสบายดีที่สุด รูปที่ 19(2) แสดงทิศทางของเซลกลางที่ได้จากการคำนวณสมการที่ 3 แล้ว โดยจะเห็นได้ว่าทิศจะเปลี่ยนเข้าหาเซลที่มีค่าความสบาย

$$Angle_i = W_{avg} * averageAngle_i + W_{com} * AngleToComfort_i \quad (8)$$

เมื่อ

$Angle_i$ คือ ค่าทิศทางใหม่ของเซล i ที่คำนวณได้จากสมการที่ 8

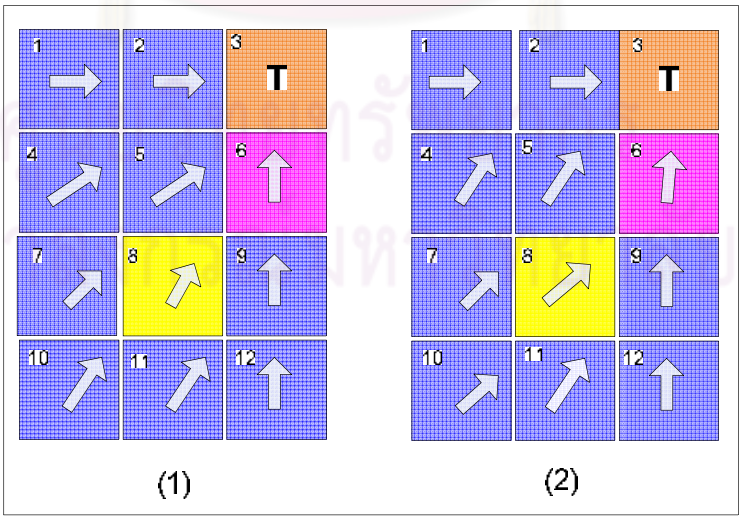
$averageAngle_i$ คือ ค่าทิศทางเฉลี่ยของเซล i ที่คำนวณจากทุกเซลที่อยู่ข้างเคียง

$AngleToComfort_i$ คือ ค่าทิศทางของเซล i ที่เคลื่อนที่ไปยังเซลที่มีค่าความสบายน้อยที่สุด

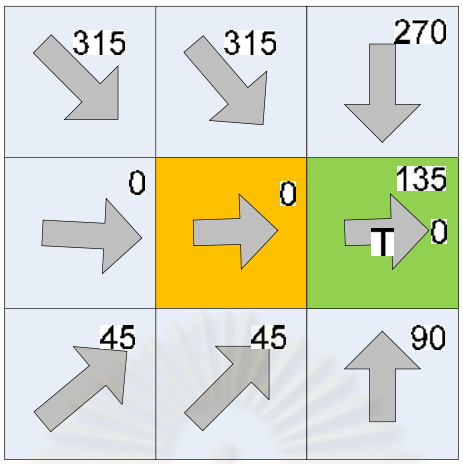
W_{avg} คือ ค่าน้ำหนัก ซึ่งในวิทยานิพนธ์นี้ให้มีค่าเท่ากับ 0.1 (จากผลการทดลอง)

W_{com} คือ ค่าน้ำหนัก ซึ่งในวิทยานิพนธ์นี้ให้มีค่าเท่ากับ 0.9 (จากผลการทดลอง)

ถ้าไม่มีการกำหนดค่าน้ำหนักให้กับสมการที่ 8 ค่ามุมที่ได้จะมีค่าผิดเพี้ยนไปในบางกรณีดังรูปที่ 20 สมมติค่ามุมตามรูปเซลที่มีสีเขียวคือเซลเป้าหมายซึ่งมีค่าความสบายน้อยที่สุด และเซลสีเหลืองคือเซลปัจจุบัน ส่วนเซลสีฟ้าเป็นเซลที่อยู่ข้างเคียง จะเห็นผลบวกของมุมทั้งหมดมีค่าเท่ากับ 1080 ซึ่งเมื่อนำมาหารด้วย 8 แล้วมีค่าเท่ากับ 135 ซึ่งถ้าให้ค่าน้ำหนักกับมุมเฉลี่ยมากเกินไปจะทำให้ค่ามุมที่ได้ผิดเพี้ยนไปซึ่งถ้าเรานำค่า 135 มาลองคูณกับค่าที่เราใช้ในการถ่วงน้ำหนักเมื่อค่าของมุมที่ได้จากเซลที่มีค่าความสบายน้อยที่สุดมีค่าเท่ากับ 0 ถ้าค่าที่ถ่วงมีค่าเป็น 0.5 จะได้มุมใหม่มีเท่ากับ 67.25 ซึ่งค่ามุมที่ได้จะทำให้เอเจนต์เคลื่อนที่ไม่เข้าสู่เป้าหมาย และถ้ากำหนดค่าน้ำหนักเป็น 0.4 จะได้ค่ามุมเป็น 54 ถ้ากำหนดให้ค่าน้ำหนักเป็น 0.3 จะได้ค่ามุมเป็น 40.5 ถ้ากำหนดให้ค่าน้ำหนักเป็น 0.2 จะได้ค่ามุมเป็น 27 แต่ถ้ากำหนดค่าน้ำหนักเป็น 0.1 จะได้ค่ามุมเป็น 13.5 ซึ่งค่ามุมของค่าน้ำหนักกับ 0.1 ทำค่าที่ได้นี้มีค่าไม่เกินค่าขอบเขตของมุมที่เรากำหนดไว้ในรูปที่ 18 เลยจึงใช้ค่าน้ำหนักเป็น 0.1 แต่ถ้าเรากำหนดค่าน้ำหนักเป็นค่าอื่นที่มากกว่า 0.1 จะทำให้โอกาสที่เอเจนต์เคลื่อนที่เข้าหาเป้าหมายที่ถูกต้องน้อยลงซึ่งอาจเกิดจากการที่เมื่อนำมุมที่ได้จากการเฉลี่ยแล้วไปรวมกับมุมที่ของเซลที่มีค่าความสบายน้อยที่สุดแล้วทำให้ค่ามุมที่ได้มีค่ามากเกินกว่าค่ามุมที่ควรจะเป็น



รูปที่ 19 แสดงการปรับทิศทางของเซลโดยสมการ 3



รูปที่ 20 แสดงค่ามุมของเซลแต่ละเซลที่ใช้ในการคำนวน

ความเร็ว คือระยะในการเคลื่อนที่ของเอเจนต์ต่อหนึ่งเฟรมของอนิเมชัน ซึ่งในโปรแกรมกำหนดค่าเริ่มต้นในสถานะแรกดังนี้ ถ้าเซลนั้นมีระยะห่างจากเป้าหมายน้อยกว่า 1.42 มีความเร็วเท่ากับ 0.5 และถ้าเซลนั้นมีระยะห่างอยู่ในช่วง 1.42 ถึง 3.64 มีความเร็วเท่ากับ 0.4 ถ้าเซลนั้นมีระยะห่างจากเป้าหมายมากกว่านี้จะมีความเร็วเป็น 0.3 ส่วนความเร็วที่เซลให้กับเอเจนต์ที่อยู่ในเซลนั้นหาได้จากการหาค่าเฉลี่ยของความเร็วของเซลที่อยู่ข้างเคียงทั้ง 8 เซล โดยที่ความเร็วสูงสุดที่สามารถใช้ในการเคลื่อนที่ของเอเจนต์ที่กำหนดไว้มีค่าน้อยกว่าหรือเท่ากับ 0.6

3.3 การชนกันของเอเจนต์

หลังจากเซลทุกเซลทำการหาความเร็วและทิศทางเสร็จสิ้น จะทำการสั่งการไปยังเอเจนต์ภายในเซลให้เคลื่อนที่ไปโดยใช้ความเร็วและทิศทางที่กำหนด เมื่อเอเจนต์เคลื่อนที่ไปอาจเกิดการชนขึ้น แม้ว่าในระดับเซลนั้น คำสั่งจากเซลจะสามารถให้เอเจนต์หลีกเลี่ยงเซลที่มีเอเจนต์อยู่เยอะแล้วก็ตาม แต่ยังไม่สามารถรับประกันได้ว่าเอเจนต์จะไม่ชนกันเองภายในเซล ในงานวิทยานิพนธ์นี้ได้มีการพยายามทดลองใช้เทคนิคในการหลีกเลี่ยงการชนในระดับตัวละคร โดยให้เอเจนต์แต่ละตัวมีตัวตรวจจับสภาวะแวดล้อมที่เอเจนต์มองเห็นทางด้านหน้า ซึ่งมีขอบเขตที่จำกัด เมื่อเอเจนต์ตรวจพบสิ่งกีดขวางจะทำการค้นหาเส้นทางที่จะเคลื่อนที่ใหม่ (เบี่ยงซ้าย เบี่ยงขวา หรือหยุดอยู่กับที่) แต่ผลการทดลองเบื้องต้นปรากฏว่าการทำงานเป็นไปอย่างช้ามาก ใช้เวลามากกว่าการทำฟลอกกิงธรรมดา ทั้งนี้อาจเป็นเพราะการคำนวณในระดับเอเจนต์นั้นมีการใช้ซีพียู

อย่างมากกว่าการคำนวณในระดับเซลอย่างมาก แม้ว่าจะลดการคำนวณลงจากสามอย่างลงมาเป็นหนึ่งอย่างก็ยังเห็นความแตกต่างได้มาก อีกทั้งการใช้เซลลูลาร์ออโตมาตายังมีการคำนวณในส่วนการสื่อสารระหว่างเซลต่างๆอีกด้วย ทำให้เกิดโอเวอร์เฮด ทำให้เวลาในการทำงานที่ได้นั้นไม่ได้เปรียบการทำงานแบบฟลอกกิงธรรมดาแต่อย่างใดถ้ามีการคิดส่วนการหลีกเลี่ยงการชนอยู่ จากผลที่ได้ทำให้ไม่สามารถใช้การหลีกเลี่ยงการชนแบบฟลอกกิงธรรมดาได้ ดังนั้นในงานวิทยานิพนธ์นี้ จึงตัดการคำนวณในส่วนนี้ออกไปและมุ่งเน้นการทดสอบผลการทำฟลอกกิงในระดับระหว่างเซลเพียงอย่างเดียวเท่านั้น

บทที่ 4

การทดลองและผลการทดลอง

4.1 วิธีการทดลอง

การทดลองนั้นเราได้นำสถานการณ์จำลองจากภาพยนตร์มาเป็นต้นแบบ สถานการณ์จำลองนั้นใช้เพื่อเปรียบเทียบสถานการณ์ที่เราสร้างขึ้นโดยใช้หลักการของฟลอกกิงตามปกติกับสถานการณ์ที่เราสร้างขึ้นโดยใช้หลักการเซลลูลาร์ฟลอกกิงที่นำเสนอในวิทยานิพนธ์นี้

สถานการณ์ฟลอกกิงตามปกติ(มัลติเอเจนต์ฟลอกกิง)นั้น เอเจนต์แต่ละตัวจะทำการค้นหาเป้าหมายเพื่อหาทิศทางที่จะเคลื่อนที่ไป รวมทั้งเปลี่ยนทิศทางและความเร็วด้วยตนเองตามหลักการฟลอกกิงของเรโนลด์[3] ในการวัดผลนั้น เราจะทำการวัดผลทั้งในด้านประสิทธิภาพในการทำงานและคุณภาพของอนิเมชัน

สำหรับประสิทธิภาพการทำงานนั้น วัดโดยเปรียบเทียบเวลาที่ใช้ในการสร้างสถานการณ์แต่ละสถานการณ์ จากการใช้เซลลูลาร์ฟลอกกิงและฟลอกกิงตามปกติ นอกจากนี้ยังต้องเปรียบเทียบการใช้งานหน่วยประมวลผลและหน่วยความจำของแต่ละสถานการณ์ เพื่อให้แน่ใจได้ว่าไม่มีกรณีที่เป็นข้อยกเว้นเกิดขึ้น

ในด้านคุณภาพนั้น วัดโดย ให้กลุ่มตัวอย่าง สังเกตการเคลื่อนที่ของตัวละครในระยะจำกัด 400 เฟรม จากสถานการณ์ภาพยนตร์ต้นฉบับ สถานการณ์ที่สร้างด้วยฟลอกกิงแบบธรรมดาและสถานการณ์ที่สร้างโดยวิธีที่นำเสนอ ให้กลุ่มตัวอย่างให้คะแนนความเหมือน (คะแนนมีระหว่าง 0 ถึง 10) ในแบบสอบถาม ซึ่งก่อนที่กลุ่มตัวอย่างจะดูวีดีโอและทำแบบสอบถามนั้น ผู้วิจัยได้ทำการบอกข้อตกลงเบื้องต้นว่า งานที่ทำขึ้นนี้ทำเพื่อสร้างต้นแบบสำหรับใช้ในการจำลองฝูงชนให้เคลื่อนที่ไปในเป้าหมายที่กำหนดไว้ ซึ่งเป้าหมายมีได้ตั้งแต่ 1 ถึง 5 เป้าหมาย และมีจำนวนตัวละครสูงสุดที่จุได้ 1600 ตัว ใช้ปิรามิดกลับหัวแทนตัวละคร พื้นที่มีสีชมพูคือเป้าหมายที่จะให้ตัวละครเคลื่อนที่ไป ส่วนการให้คะแนนนั้นมีหลักเกณฑ์ดังนี้ ให้ดูว่ารูปแบบของกลุ่มตัวละครในฉากภาพยนตร์มีความเหมือนกับสถานการณ์จำลองเพียงใด และให้คะแนนความเหมือนของสถานการณ์จำลองทั้ง 2 ที่สร้างขึ้นมาเลียนแบบนั้น โดยสามารถให้คะแนนได้ตั้งแต่ 1 ถึง 10 โดยที่การให้คะแนนเรียงจากน้อยไปมาก คะแนนเท่ากับ 1 คือคะแนนน้อยสุด คะแนนเท่ากับ 5 คือปาน

กลาง ส่วน คะแนนเท่ากับ 10 คือ เหมือนมาก และให้กลุ่มทดลองดูวีดีโอโดยมีการเรียงวีดีโอ ดังนี้ เริ่มจาก สถานการ์ต้นแบบนั้น ต่อด้วยวีดีโอของเซลลูลาร์ฟลอกกิง และตามด้วยมัลติเอเจนต์ฟลอกกิง เมื่อดูครบทั้ง 3 วีดีโอแล้วจะให้กลุ่มตัวอย่างให้คะแนน เมื่อให้คะแนนเสร็จจึงจะให้ดูสถานการณ์ต่อไป ทำอย่างนี้ไปจนครบ 10 สถานการณ์ ซึ่งกลุ่มตัวอย่างเกือบครึ่งหนึ่ง มีความรู้เกี่ยวกับหลักการของฟลอกกิงไม่มากก็น้อย การสอบถามจากกลุ่มเช่นนี้มีข้อดีคือ กลุ่มตัวอย่างนี้สามารถทำการสังเกตเฉพาะในส่วนที่ทางผู้วิจัยต้องการได้จริง ไม่นำเอาส่วนในภาพยนตร์ที่ไม่เกี่ยวข้องกับงานวิทยานิพนธ์นี้มาคิด เนื่องจากทราบว่าการพฤติกรรมที่สำคัญของฟลอกกิงเป็นเช่นไร แต่ในขณะเดียวกันก็มีข้อเสียเพราะอาจมุ่งเน้นพฤติกรรมหลักจนเกินไปจนเกิดความลำเอียงในการให้คะแนนความเหมือนได้

4.2 การทำงานของสถานการณ์ที่ใช้อัลกอริธึมฟลอกกิงแบบปกติ

เอเจนต์จะถูกสร้างขึ้นตามตำแหน่งที่กำหนดไว้โดยเลือกจากขอบเขตของพื้นที่ที่กำหนดให้ ผู้ควบคุมต้องทำการกำหนดจุดหมายในการเคลื่อนที่ให้กับเอเจนต์ ทำการกำหนดความเร็วเริ่มต้นให้กับเอเจนต์โดยดูจากจำนวนเอเจนต์ที่อยู่รอบข้าง จากนั้นโปรแกรมจะทำการหาทิศทางในตอนเริ่มต้นโดยตรวจหาว่าจุดหมายในการเคลื่อนที่อยู่ทิศทางใดของเอเจนต์ ซึ่งจะได้มุมในการเคลื่อนที่สู่จุดหมายมาโดยตรงจากนั้นโปรแกรมจะทำการปรับความเร็วกับทิศทางให้กับเอเจนต์แต่ละตัวให้สอดคล้องกับเอเจนต์ที่อยู่รอบข้าง (โดยพยายามปรับเข้าหาความเร็วเฉลี่ยและทิศทางเฉลี่ยจากเอเจนต์รอบข้าง) การปรับความเร็วนี้จะใช้ตลอดการเคลื่อนที่ของเอเจนต์ทุกตัว เมื่อเอเจนต์แต่ละตัวเคลื่อนที่ไปจะทำการตรวจสอบตำแหน่งที่ต้องเคลื่อนที่ไปว่าสามารถเคลื่อนที่ไปได้หรือไม่และทำการหลบหลีกสิ่งกีดขวางโดยอาศัยเซ็นเซอร์ในการตรวจสอบสภาวะของพื้นที่รอบข้างของตัวเอเจนต์ เอเจนต์แต่ละตัวต้องคอยระวังไม่ให้ออกห่างจากตัวอื่นมากเกินไป ในตัวโปรแกรมได้กำหนดเอาไว้ว่าเมื่อเอเจนต์เคลื่อนที่ไปถึงเป้าหมายแล้ว เอเจนต์จะเคลื่อนที่ต่อไปเพื่อออกจากหน้าจอ ทั้งนี้เพื่อให้เอเจนต์ตัวอื่นสามารถเคลื่อนที่เข้ามายังจุดหมายได้ด้วย

รหัสเทียมของอัลกอริทึมฟลอกกิง[20]

การสั่งการให้เอเจนต์ทุกตัวเคลื่อนที่ต้องอาศัยกฎของฟลอกกิงทั้ง 3 ข้อ โดยที่กฎแต่ละข้อเป็นอิสระต่อกัน ซึ่งในกฎแต่ละข้อจะมีการใช้เวกเตอร์อย่างง่ายในการคำนวณ ที่ตำแหน่งของเอเจนต์ (บอย) แต่ละตัว เพื่อนำมากำหนดตำแหน่งให้กับเอเจนต์แต่ละตัว ดังรูปที่ 21

```
PROCEDURE move_all_boids_to_new_positions()
        Vector v1, v2, v3
        Boid b
        FOR EACH BOID b
                v1 = rule1(b)
                v2 = rule2(b)
                v3 = rule3(b)
                b.velocity = b.velocity + v1 + v2 + v3
                b.position = b.position + b.velocity
        END
END PROCEDURE
```

รูปที่ 21 แสดงรหัสเทียมของการเคลื่อนที่ของบอยโดยรวม [20]

กฎข้อที่ 1 (rule1) การหลีกเลี่ยงการเบียดเสียดจากตัวข้างเคียง หาโดยกำหนดเวกเตอร์ศูนย์ ทำการเปรียบเทียบระยะห่างของเอเจนต์ปัจจุบันกับเอเจนต์ที่อยู่รอบข้างมันทุกตัว แล้วดูว่าค่าระยะห่างตัวไหนมีค่าน้อยกว่าระยะที่สามารถเข้าใกล้ได้มากที่สุด (ในโค้ดนี้มีค่าเท่ากับ 100) ซึ่งสำหรับงานเรานั้นใช้ค่า 2.5 แทน ถ้าระยะห่างระหว่างเอเจนต์มีค่าน้อยกว่าระยะที่กำหนดจะนำเวกเตอร์ c มาลบกับผลต่างนั้น เมื่อทำการคำนวณแต่ละตัวจะส่งค่าเวกเตอร์ c นี้กลับไปเพื่อรวมกับความเร็วของตัวมันเอง ดังรูปที่ 22

```
PROCEDURE rule1(boid bJ)
        Vector c = 0;
        FOR EACH BOID b
                IF b != bJ THEN
                        IF |b.position - bJ.position| < 100 THEN
                                c = c - (b.position - bJ.position)
                        END IF
                END IF
        END
        RETURN c
END PROCEDURE
```

รูปที่ 22 แสดงรหัสเทียมของกฎข้อที่ 1 [20]

กฎข้อที่ 2 การเคลื่อนที่ไปในทิศทางใกล้เคียงกับตัวที่อยู่ข้างเคียง หาโดยการกำหนดเวกเตอร์ PV_j คือเวกเตอร์ความเร็ว ในโค้ดตัวอย่างในรูปที่ 23 จะทำการหาผลรวมของเวกเตอร์ความเร็วของเอเจนต์แต่ละตัว แล้วนำมาหาค่าเฉลี่ยตามจำนวนตัวที่อยู่ข้างเคียง จากนั้นหาผลต่างระหว่างค่าเฉลี่ยที่ได้มากับความเร็วของตัวมันเอง แล้วหารด้วย 8 จากนั้นจึงส่งค่าที่ได้กลับ

```
PROCEDURE rule2(boid bJ)
        Vector pvJ
        FOR EACH BOID b
                IF b != bJ THEN
                        pvJ = pvJ + b.velocity
                END IF
        END
        pvJ = pvJ / N-1
        RETURN (pvJ - bJ.velocity) / 8
END PROCEDURE
```

รูปที่ 23 แสดงรหัสเทียมของกฎข้อที่ 2 [20]

กฎข้อที่ 3 การรักษาระยะห่างจากตัวรอบข้าง ทำโดยหาผลรวมของตำแหน่งของตัวที่อยู่รอบข้างมันทั้งหมด จากนั้นนำค่าเฉลี่ยมาหารด้วย 100 ค่า 100 คือค่าเปอร์เซ็นที่เอเจนต์จะเข้าสู่ศูนย์กลางกลุ่ม ดังรูปที่ 24

```
PROCEDURE rule3(boid bJ)
        Vector pcJ
        FOR EACH BOID b
                IF b != bJ THEN
                        pcJ = pcJ + b.position
                END IF
        END
        pcJ = pcJ / N-1
        RETURN (pcJ - bJ.position) / 100
END PROCEDURE
```

รูปที่ 24 แสดงรหัสเทียมของกฎข้อที่ 3 [20]

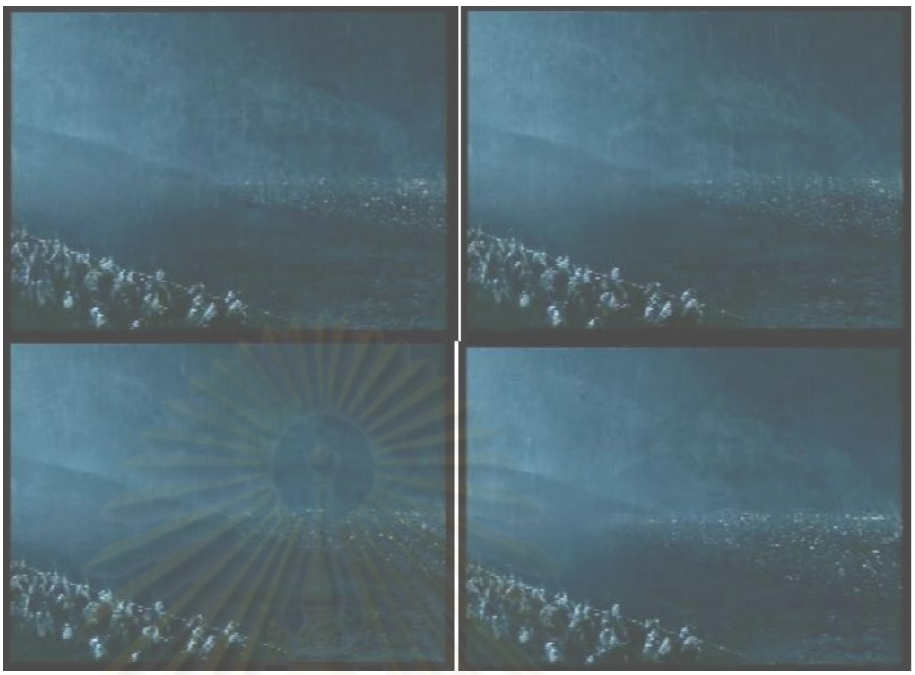
4.3 สถานการณ์จำลองจากภาพยนตร์

สถานการณ์จากภาพยนตร์ โฆษณา และเกม ได้ถูกเลือกมา 10 สถานการณ์ ซึ่งเราได้เลือกสถานการณ์เหล่านี้มาเพื่อทดสอบว่า ผลที่ได้จากโปรแกรมต้นแบบที่เราได้สร้างขึ้นนั้นสามารถแสดงพฤติกรรมได้คล้ายคลึงเพียงใด ซึ่งสถานการณ์จำลองเหล่านี้ถูกสร้างขึ้นโดยใช้โปรแกรมตัวเดียวกันแต่มีการผสมผสานคอมพิวเตอร์กราฟิกกับโมชันแคบเจอร์ลงไปด้วย

สถานการณ์ที่ 1 นำมาจากภาพยนตร์เรื่องนาร์เนีย เป็นการเคลื่อนที่ของกลุ่มคนที่อยู่ด้านบนซ้ายเคลื่อนที่จากด้านบนซ้ายไปทางด้านล่างขวา ฝูงชนมีลักษณะรูปสามเหลี่ยม ดังรูปที่ 25

รูปที่ 25 แสดงภาพการเคลื่อนที่โดยรวมของสถานการณ์จำลองที่ 1

สถานการณ์ที่ 2 นำมาจากภาพยนตร์เรื่องลอร์ด ออฟเดอริง ภาค 2 เป็นการเคลื่อนที่ของฝูงชนกลุ่มขนาดใหญ่เคลื่อนที่จากด้านขวาเคลื่อนที่เข้าหาฝูงคนที่อยู่ทางด้านซ้ายล่างของแผนที่ ฝูงชนมีลักษณะเรียงหน้ากระดานที่มีส่วนยื่นออกทำให้เป็นรูปร่างคล้ายสี่เหลี่ยมคางหมู ดังรูปที่ 26



รูปที่ 26 แสดงภาพการเคลื่อนที่โดยรวมของสถานการณ์จำลองที่ 2

สถานการณ์ที่ 3 นำมาจากภาพยนตร์เรื่องนาร์เนีย เป็นการเคลื่อนที่เข้าหากันของกลุ่มคน ด้านหนึ่งเคลื่อนที่จากด้านซ้ายไปด้านขวาส่วนอีกกลุ่มหนึ่งเคลื่อนที่จากทางด้านขวาไปทางด้านซ้าย ทั้ง 2 กลุ่มเข้ามาพบกันตรงกลางแผนที่ดังรูป 27 ซึ่งได้จำลองฝูงคนที่เคลื่อนที่จากซ้ายมาทางขวา ฝูงชนมีลักษณะเป็นรูปสามเหลี่ยมค่อนข้างแคบ

รูปที่ 27 แสดงภาพการเคลื่อนที่โดยรวมของสถานการณ์จำลองที่ 3

สถานการณ์ที่ 4 นำมาจากภาพยนตร์เรื่องนาร์เนีย เป็นการเคลื่อนที่ของฝูงชนจากทางด้านบนซ้ายเคลื่อนที่ไปทางด้านล่างซ้ายของแผนที่ ฝูงชนมีลักษณะเป็นแถวแคบที่มีจำนวนคนไม่มาก เดินไม่เป็นระเบียบนัก ดังรูปที่ 28

รูปที่ 28 แสดงภาพการเคลื่อนที่โดยรวมของสถานการณ์จำลองที่ 4

สถานการณ์ที่ 5 นำมาจากภาพยนตร์เรื่อง ลอร์ด ออฟเดอริง ภาค 3 เป็นการเคลื่อนที่ของฝูงชนจากทางด้านซ้ายเคลื่อนที่ไปทางด้านขวาของแผนที่ ฝูงชนมีลักษณะเป็นแถวตอนเดินเป็นระเบียบ ความกว้างน้อยดังรูปที่ 29

รูปที่ 29 แสดงภาพการเคลื่อนที่โดยรวมของสถานการณ์จำลองที่ 5

สถานการณ์ที่ 6 นำมาจากภาพยนตร์เรื่อง ขงจื้อ เป็นการเคลื่อนที่ของฝูงคนกลุ่มใหญ่จากทางด้านบนขวาเคลื่อนที่ไปยังด้านบนซ้ายของแผนที่ ฝูงชนมีลักษณะเป็นหลายแถวตอนเคลื่อนที่ไปในทิศทางเดียวกันคล้ายการเดินสวนสนามของทหาร ดังรูปที่ 30

รูปที่ 30 แสดงภาพการเคลื่อนที่โดยรวมของสถานการณ์จำลองที่ 6

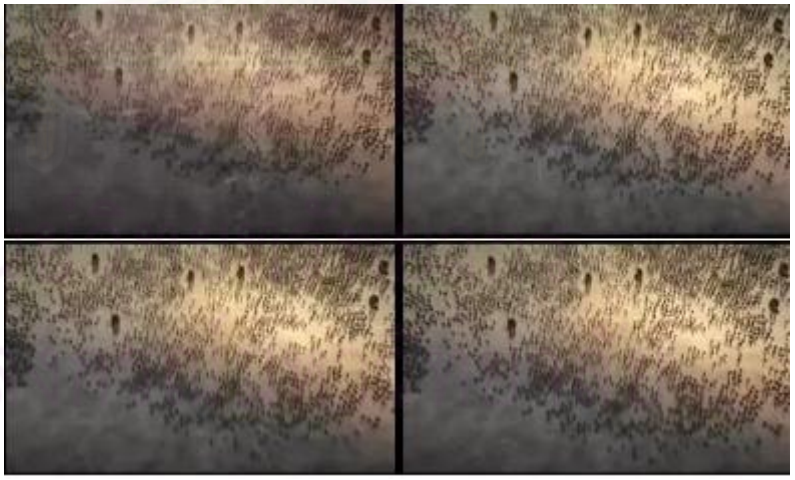
สถานการณ์ที่ 7 นำมาจากภาพยนตร์เรื่อง ขงจื้อ เป็นการเคลื่อนที่ของฝูงชนกลุ่มใหญ่เคลื่อนจากทางด้านบนซ้ายเคลื่อนที่ไปทางด้านล่างขวา ฝูงชนมีลักษณะเป็นกลุ่มใหญ่เคลื่อนที่ไปในทิศทางเดียวกัน ดังรูปที่ 31

รูปที่ 31 แสดงภาพการเคลื่อนที่โดยรวมของสถานการณ์จำลองที่ 7

สถานการณ์ที่ 8 นำมาจากภาพยนตร์เรื่อง ขงจื้อ เป็นการเคลื่อนที่ของฝูงชนเคลื่อนจากทางด้านขวาไปทางด้านซ้าย หน้าฝูงชนมีลักษณะเป็นสามเหลี่ยมสองรูป ดังรูปที่ 32

รูปที่ 32 แสดงภาพการเคลื่อนที่โดยรวมของสถานการณ์จำลองที่ 8

สถานการณ์ที่ 9 นำมาจากภาพยนตร์เรื่อง ขงจื้อ เป็นการเคลื่อนที่ของฝูงชนขนาดใหญ่เคลื่อนจากด้านบนไปยังด้านล่างของแผนที่ ฝูงชนมีลักษณะเรียงตัวไม่เป็นระเบียบเคลื่อนที่ไปในทิศทางเดียวกัน ดังรูปที่ 33



รูปที่ 33 แสดงภาพการเคลื่อนที่โดยรวมของสถานการณ์จำลองที่ 9

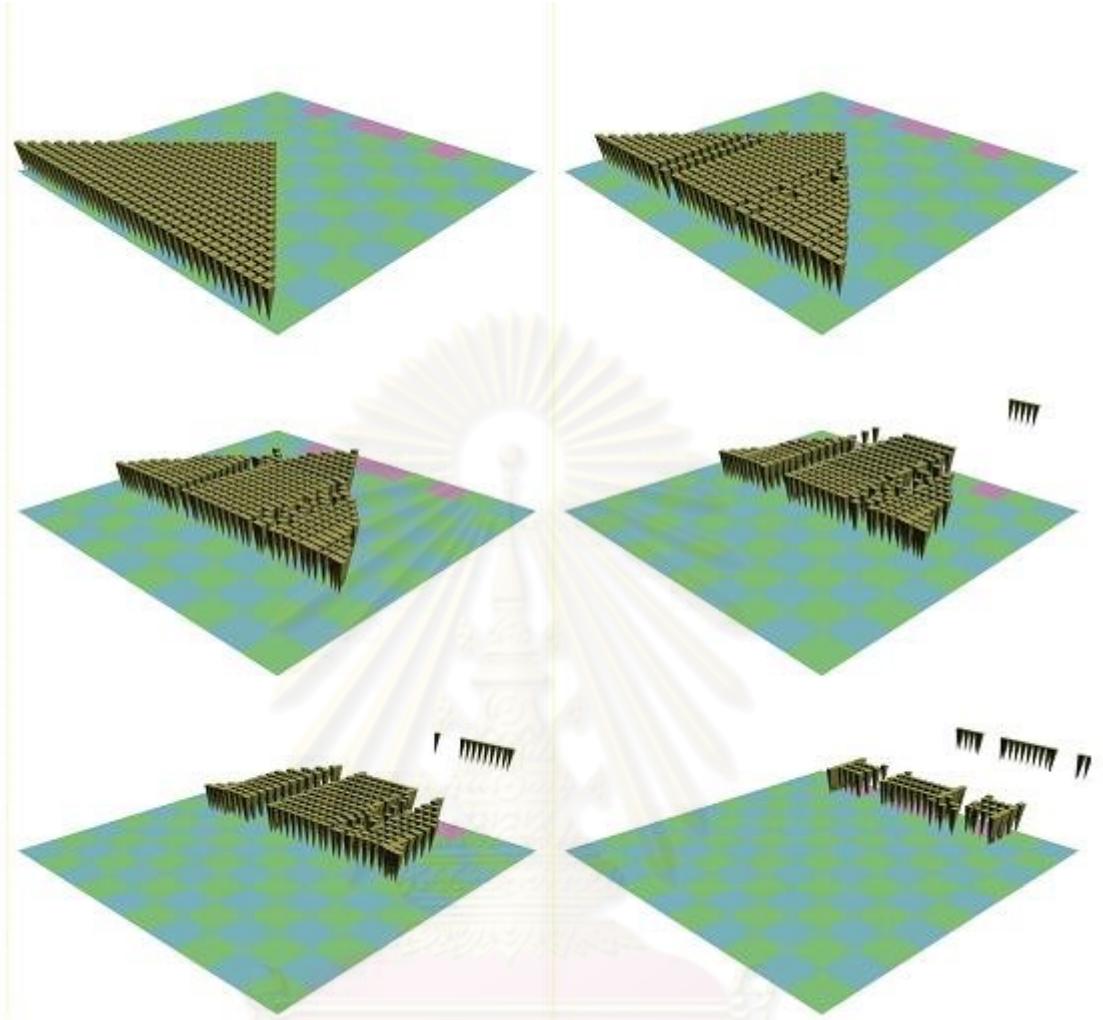
สถานการณ์ที่ 10 นำมาจากภาพยนตร์เรื่อง มัมมี่ 3 เป็นการเคลื่อนที่ของกองทหารโครงกระดูก เคลื่อนที่ไปหากลุ่มทหารดินเผาเป็นการเคลื่อนที่จากทางด้านซ้ายไปด้านขวาโดยกลุ่มทหารแบ่งออกเป็นสามกลุ่มย่อย ดังรูปที่ 34

รูปที่ 34 แสดงภาพการเคลื่อนที่โดยรวมของสถานการณ์จำลองที่ 10

4.4 ผลการทดลอง

4.4.1. พฤติกรรมที่ได้จากเซลลูลาร์ฟลอกกิง

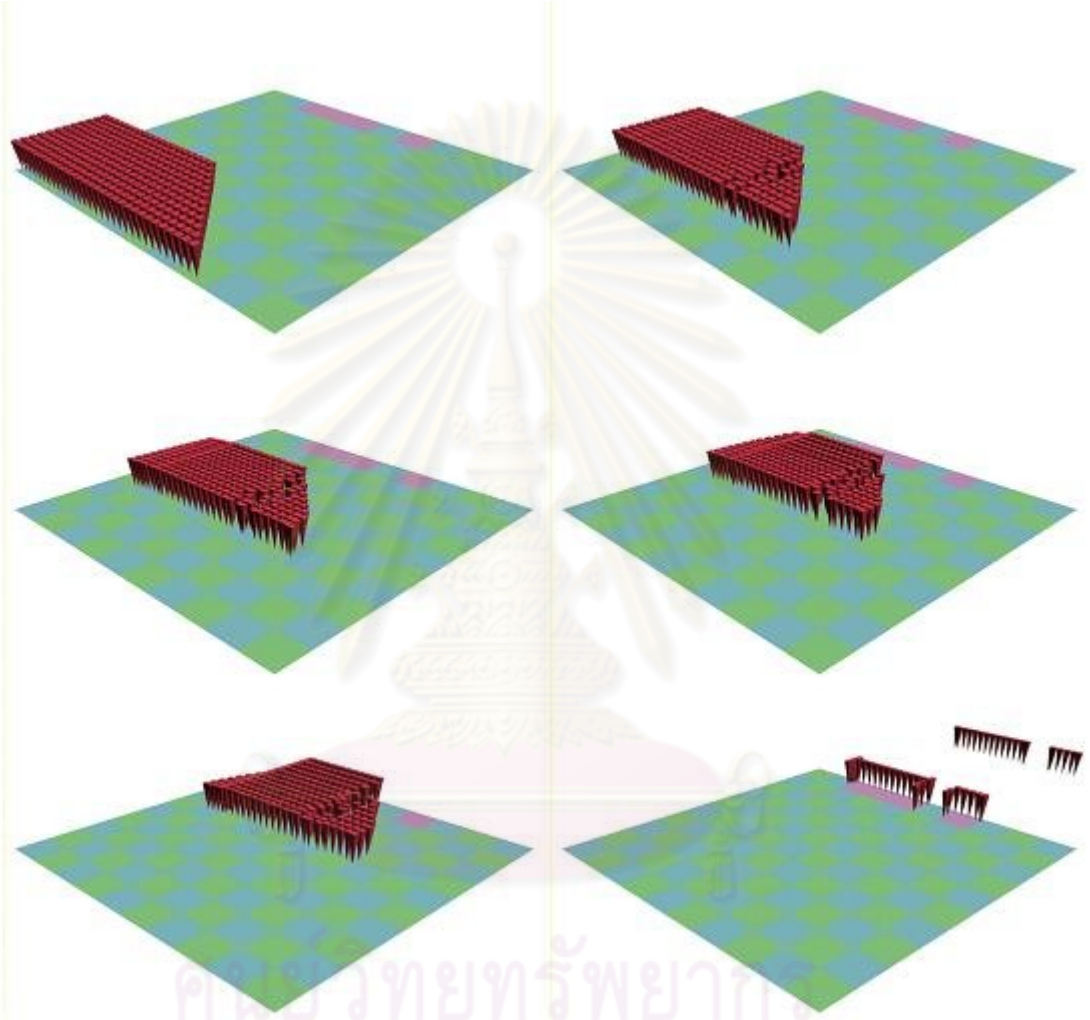
ผลของโปรแกรมที่สร้างเลียนแบบสถานการณ์จำลองที่ 1 ซึ่งได้รับรูปแบบของฝูงคนจากไฟล์ ซึ่งฝูงชนนี้เคลื่อนที่จากทางด้านซ้ายมือไปทางด้านขวามือ มีเซลเป้าหมายที่กำหนดให้ 4 เซลคือเซลหมายเลข 30 50 60 และ 80 มีผลดังแสดงในรูปที่ 35



รูปที่ 35 แสดงภาพการเคลื่อนที่โดยรวมจากโปรแกรมในสถานการณ์ที่ 1

จากการสังเกตพฤติกรรม พบว่ามีการเดินที่ค่อนข้างเป็นระเบียบมากกว่าสถานการณ์ต้นฉบับ รูปลักษณ์การเคลื่อนที่นั้นมีความคล้ายต้นฉบับ เมื่อใกล้จุดหมายฝูงชนจะแยกตัวเพื่อไปยังจุดหมายย่อยต่างๆกัน มีเอเจนต์บางตัวที่พยายามเข้าไปเกาะกลุ่มกับตัวอื่น และดูเหมือนรวมไปกับตัวอื่น ในสถานการณ์ต้นฉบับจะไม่มีเหตุการณ์นี้ ทั้งนี้เป็นเพราะว่า เซลลูลาร์ฟลอกกิงไม่ได้มีการคิดเรื่องการชนกันของตัวละคร วิธีป้องกันปัญหาเช่นนี้โดยไม่ต้องคิดการชน อาจทำได้โดย กำหนดให้เซลแต่ละเซลมีขนาดใหญ่ และให้จุจำนวนคนได้น้อยกว่านี้ ซึ่งจะทำให้เกิดช่องว่างภายในเซลเพียงพอที่ตัวละครจะไม่ชนกัน

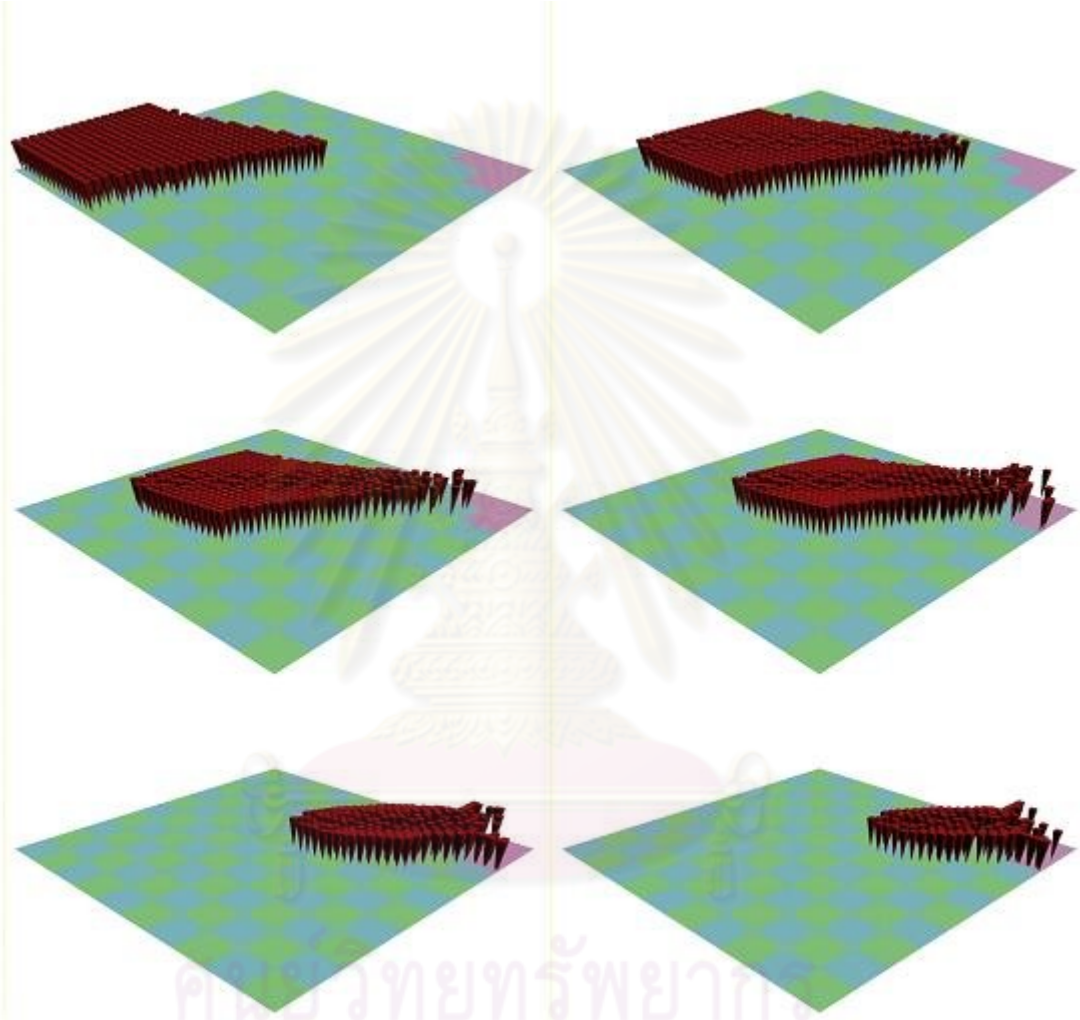
ผลของโปรแกรมที่สร้างเลียนแบบสถานการณ์จำลองที่ 2 ซึ่งได้รับรูปแบบของฝูงคนจากไฟล์ ซึ่งฝูงชนนี้เคลื่อนที่จากทางด้านซ้ายมือไปทางด้านขวามือ มีเซลเป้าหมายที่กำหนดให้ 4 เซลคือเซลหมายเลข 30 40 50 และ 70 มีผลดังแสดงในรูปที่ 36



รูปที่ 36 แสดงภาพการเคลื่อนที่โดยรวมจากโปรแกรมในสถานการณ์ที่ 2

จากการสังเกตพฤติกรรม พบว่ามีการเคลื่อนไหวที่เป็นระเบียบคล้ายกับต้นฉบับแต่พอถึงจุดหมาย (ในภาพยนตร์ตัวต้นฉบับนั้น ฝูงชนจะหยุดก่อนที่จะถึงจุดหมาย) ซึ่งมีขนาดเล็กตัวละครจะพยายามเบียดเสียดกันจนแน่นมาก ซึ่งเป็นปัญหาที่เกิดจากการไม่จัดการการชนอีกเช่นเดียวกัน

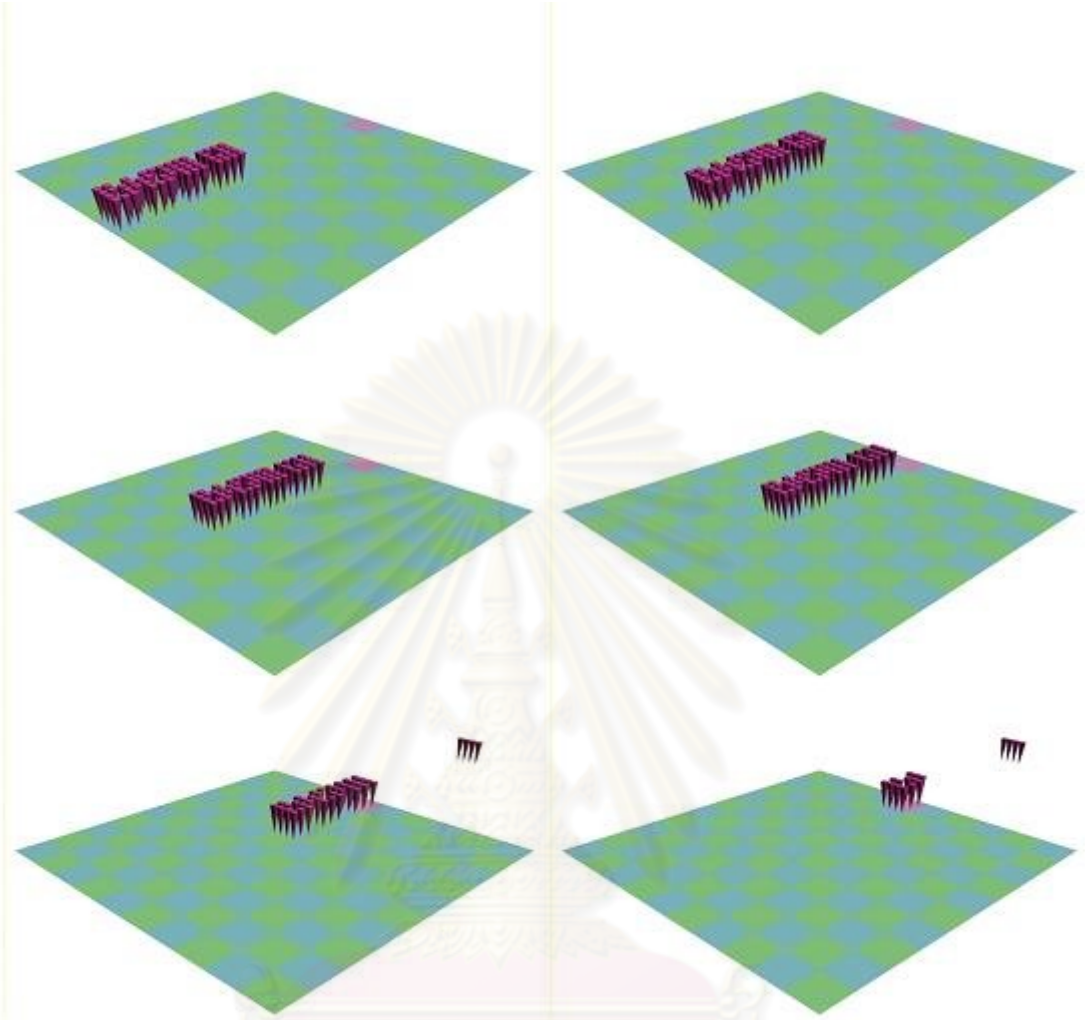
ผลของโปรแกรมที่สร้างเลียนแบบสถานการณ์จำลองที่ 3 ซึ่งได้ได้รับรูปแบบของฝูงคนจากไฟล์ ซึ่งฝูงชนนี้เคลื่อนที่จากทางด้านบนซ้ายไปทางด้านล่างขวา มีเซลเป้าหมายที่กำหนดให้ 3 เซลคือ 90, 99 และ 100 มีผลดังแสดงในรูปที่ 37



รูปที่ 37 แสดงภาพการเคลื่อนที่โดยรวมจากโปรแกรมในสถานการณ์ที่ 3

พฤติกรรมมีความคล้ายต้นฉบับอยู่มาก ตัวละครเดินได้ค่อนข้างเป็นระเบียบ ปัญหาการชนกันดูไม่ชัดนักในสถานการณ์นี้

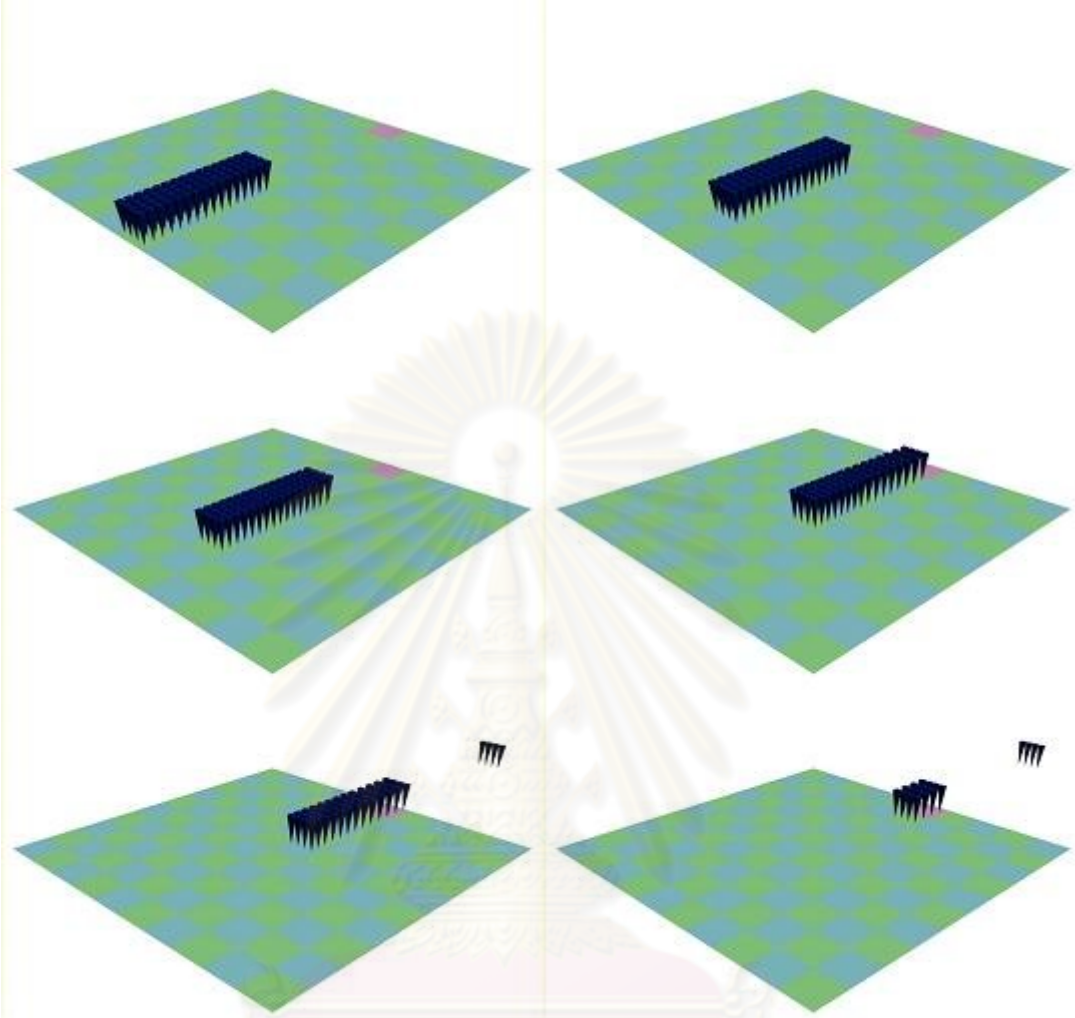
ผลของโปรแกรมที่สร้างเลียนแบบสถานการณ์จำลองที่ 4 ซึ่งได้ได้รับรูปแบบของฝูงคนจากไฟล์ ซึ่งฝูงชนนี้เคลื่อนที่จากทางด้านซ้ายไปทางด้านขวา มีเซลเป้าหมายที่กำหนดให้ 1 เซลคือเซลหมายเลข 50 มีผลดังแสดงในรูปที่ 38



รูปที่ 38 แสดงภาพการเคลื่อนที่โดยรวมจากโปรแกรมในสถานการณ์ที่ 4

ตัวละครเดินไปยังจุดหมายได้เช่นเดียวกับต้นฉบับ โดยมีการเดินที่เป็นระเบียบมาก ดูเหมือนเคลื่อนที่ไปพร้อมกันตลอด ซึ่งในต้นฉบับจะไม่เคลื่อนที่เป็นระเบียบมากขนาดนี้ ซึ่งนี่เป็นสัญญาณบอกว่า เซลลูลาร์ฟลอกกิงไม่เหมาะกับการแสดงการเคลื่อนที่ที่ไม่เป็นระเบียบโดยเฉพาะอย่างยิ่งเมื่อมีจำนวนเอเจนต์น้อย เพราะเห็นพฤติกรรมได้ชัดเจนมาก

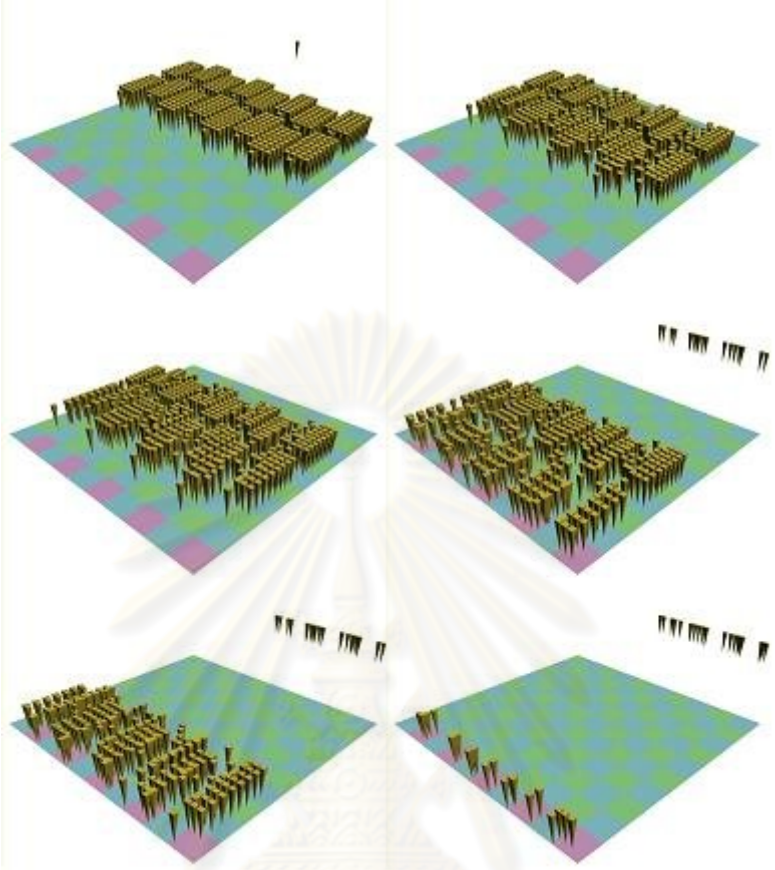
ผลของโปรแกรมที่สร้างเลียนแบบสถานการณ์จำลองที่ 5 ซึ่งได้ได้รับรูปแบบของฝูงคนจากไฟล์ ซึ่งฝูงชนนี้เคลื่อนที่จากทางด้านซ้ายไปทางด้านขวา มีเซลเป้าหมายที่กำหนดให้ 1 เซลคือเซลหมายเลข 60ได้ผลดังแสดงในรูปที่ 39



รูปที่ 39 แสดงภาพการเคลื่อนที่โดยรวมจากโปรแกรมในสถานการณ์ที่ 5

ในสถานการณ์ต้นฉบับนั้น การเคลื่อนที่เป็นระเบียบอย่างมาก ซึ่งเซลลูลาร์ฟลอกกิงก็แสดงพฤติกรรมเช่นเดียวกัน จึงสามารถสรุปได้ว่า เซลลูลาร์ฟลอกกิงสามารถใช้กับการเคลื่อนที่ที่เป็นระเบียบเป็นกลุ่มก้อนได้ดี

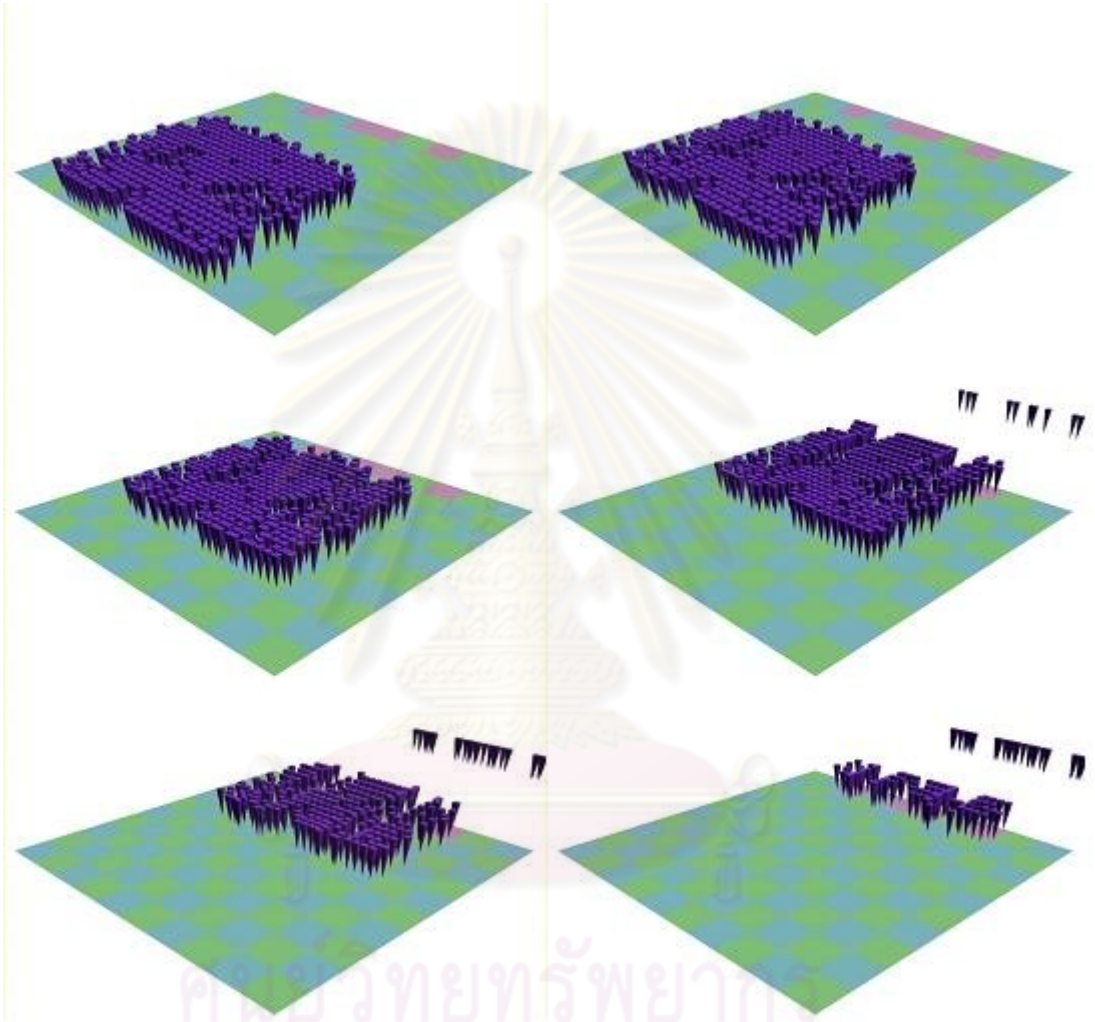
ผลของโปรแกรมที่สร้างเลียนแบบสถานการณ์จำลองที่ 6 ซึ่งได้รับรูปแบบของฝูงคนจากไฟล์ ซึ่งฝูงชนนี้เคลื่อนที่จากทางด้านขวาไปทางด้านซ้าย มีเซลเป้าหมาย 5 เซลคือ 11, 31, 51, 71 และ 91 มีผลดังแสดงในรูปที่ 40



รูปที่ 40 แสดงภาพการเคลื่อนที่โดยรวมจากโปรแกรมในสถานการณ์ที่ 6

จากการสังเกตพฤติกรรม พบว่ามีการเคลื่อนไหวที่เป็นระเบียบก็ต่อเมื่อเซลที่เอเจนต์อยู่นั้นอยู่ในแนวเดียวกันกับเซลเป้าหมายซึ่งจะคล้ายกับต้นฉบับ แต่เอเจนต์ที่เซลไม่ได้อยู่แถวเดียวกับเซลเป้าหมายนั้นจะพยายามเคลื่อนที่เข้าไปแทรกกับเซลที่อยู่ในแนวเดียวกับเป้าหมายจึงเกิดการเดินสลับไปมาจนถึงเป้าหมาย พฤติกรรมการแทรกตัวนี้สามารถแก้ไขได้โดยการกำหนดจุดเป้าหมายเพิ่มเติม จากรูป 34 ถ้าเพิ่มจุดเป้าหมายให้ครอบคลุมเซลด้านซ้ายล่างทั้งหมด ตัวละครจะเดินเป็นเส้นตรงอย่างเป็นระเบียบ ตัวที่อยู่นอกแถวก็จะไม่มีการไปรวมแถว แต่ตามปกติแล้ว ฝูงชนต้องมีความไม่เป็นระเบียบอยู่บ้าง มีการเปลี่ยนแปลงจัดตัวเองบ้างจึงจะดูสมจริง ถ้ากำหนดจุดเป้าหมายเพื่อให้เดินตรงเพียงอย่างเดียว สถานการณ์จะออกมาไม่ต่างกับการโปรแกรมเอเจนต์ให้เดินตรง พฤติกรรมการแทรกเข้าไปร่วมแถวนั้นถือว่าเป็นพฤติกรรมที่สมจริงกว่าการที่เอเจนต์ทุกตัวเดินตรง

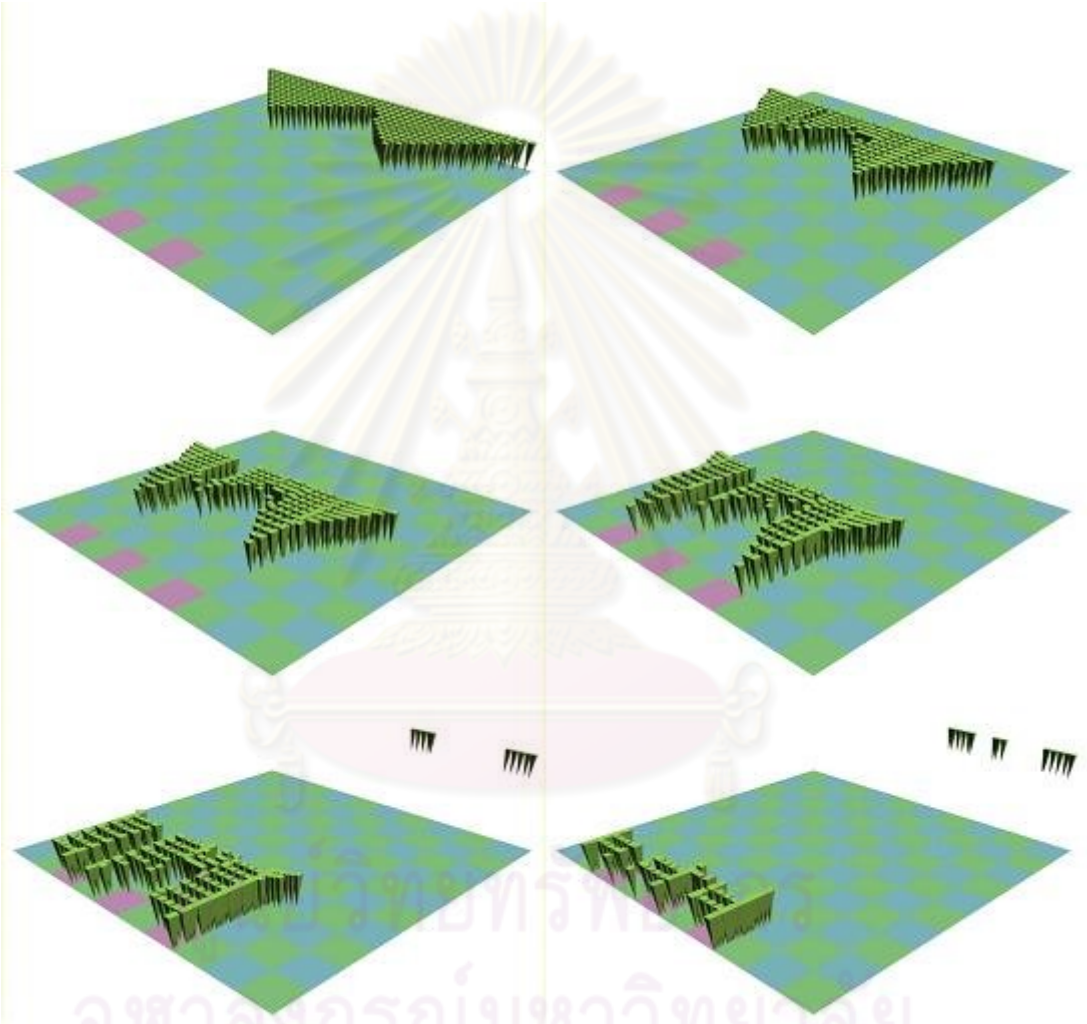
ผลของโปรแกรมที่สร้างเลียนแบบสถานการณ์จำลองที่ 7 ซึ่งได้รับรูปแบบของฝูงคนจากไฟล์ ซึ่งฝูงชนนี้เคลื่อนที่จากทางด้านซ้ายไปทางด้านขวา มีเซลเป้าหมาย 4 เซลคือ 30, 50, 60 และ 80 มีผลดังแสดงในรูปที่ 41



รูปที่ 41 แสดงภาพการเคลื่อนที่โดยรวมจากโปรแกรมในสถานการณ์ที่ 7

จากการสังเกตพฤติกรรม พบว่ามีการเคลื่อนไหวที่เป็นระเบียบคล้ายกับต้นฉบับ แต่พอใกล้ถึงจุดหมาย จะเห็นการเคลื่อนที่แบบแยกทางอย่างเห็นได้ชัดตรงเซลที่ไม่ได้อยู่แนวเดียวกับเซลเป้าหมาย ซึ่งเป็นพฤติกรรมโดยทั่วไปของเอเจนต์ที่พยายามเคลื่อนที่เข้าสู่เป้าหมาย สำหรับการชนกันนั้นยังคงมีแต่เห็นได้ไม่ชัดเท่าสถานการณ์อื่น ทั้งนี้อาจเป็นเพราะพื้นที่เป้าหมายมีขนาดใหญ่และกว้างใกล้เคียงกับความกว้างของฝูงชน ทำให้เอเจนต์แยกย้ายไปตามเป้าหมายได้มาก จึงไม่เกิดการเบียดเสียดกันมากนัก

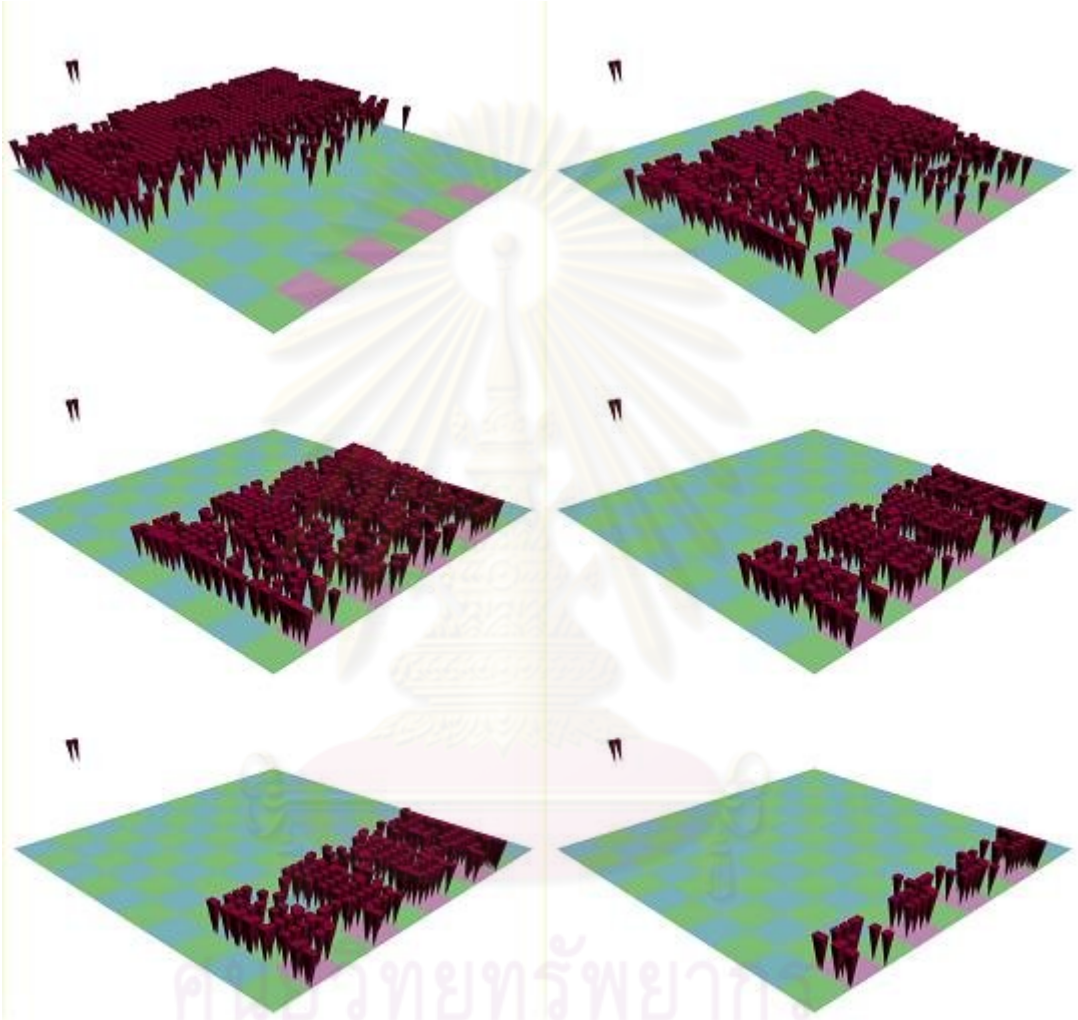
ผลของโปรแกรมที่สร้างเลียนแบบสถานการณ์จำลองที่ 8 ซึ่งได้รับรูปแบบของฝูงคนจากไฟล์ ซึ่งฝูงชนนี้เคลื่อนที่จากทางด้านขวาไปทางด้านซ้าย มีเซลเป้าหมาย 4 เซลคือ 11, 31, 51 และ 71 มีผลดังแสดงในรูปที่ 42



รูปที่ 42 แสดงภาพการเคลื่อนที่โดยรวมจากโปรแกรมในสถานการณ์ที่ 8

พบว่า เมื่อจุดหมายปลายทางมีขนาดเล็กเช่นในสถานการณ์นี้ เซลลูลาร์ฟลอกกิงจะแสดงการเบียดและรวมตัวของเอเจนต์ให้เห็น ดังนั้นเซลจุดหมายที่มีขนาดใหญ่จึงมีความสำคัญในการป้องกันพฤติกรรมการชน

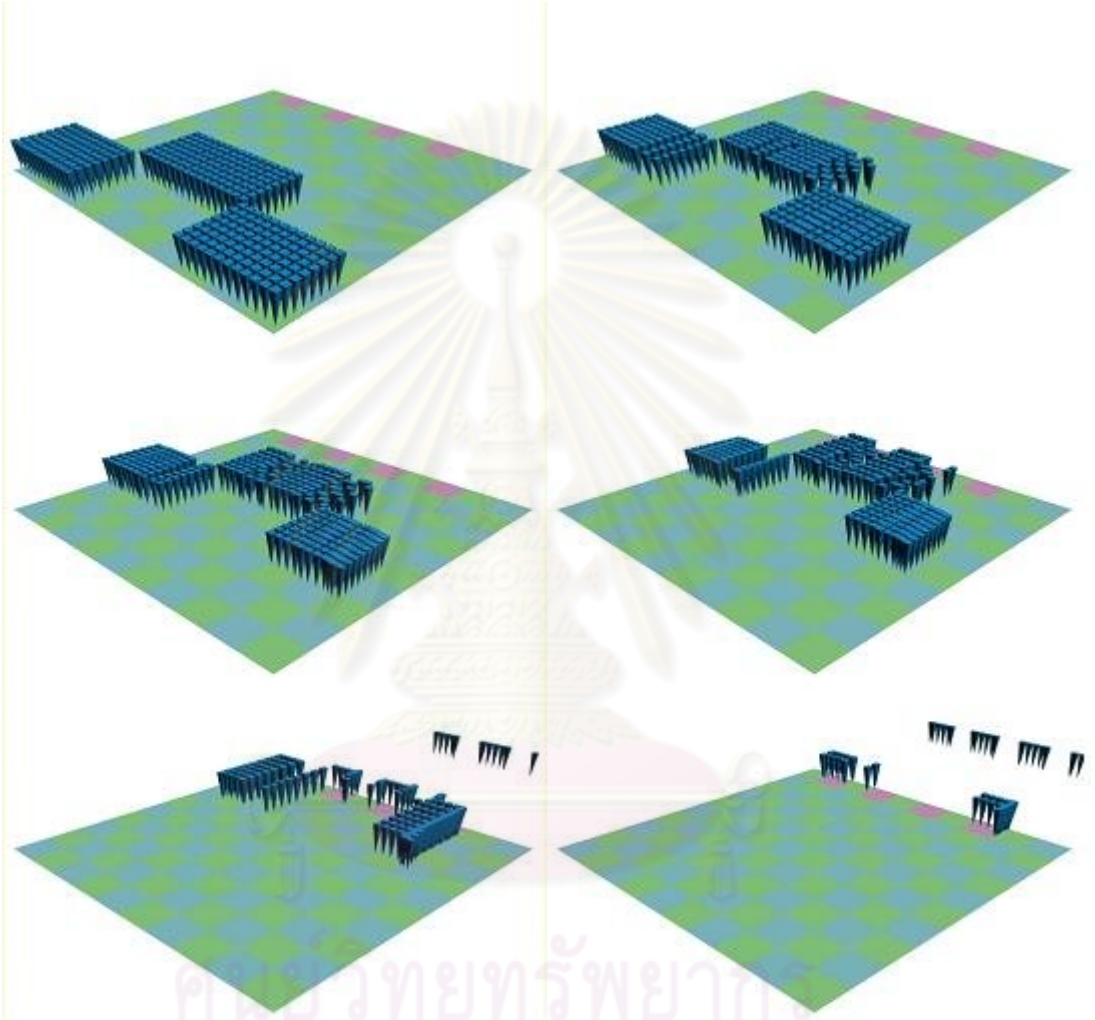
ผลของโปรแกรมที่สร้างเลียนแบบสถานการณ์จำลองที่ 9 ซึ่งได้รับรูปแบบของฝูงคนจากไฟล์ ซึ่งฝูงชนนี้เคลื่อนที่จากทางด้านบนไปทางด้านล่าง มีเซลเป้าหมาย 4 เซลคือ 92, 94, 96 และ 98 มีผลดังแสดงในรูปที่ 43



รูปที่ 43 แสดงภาพการเคลื่อนที่โดยรวมจากโปรแกรมในสถานการณ์ที่ 9

จากการสังเกตพฤติกรรม พบว่ามีการเคลื่อนไหวที่ไม่เป็นระเบียบคล้ายกับต้นฉบับ แต่พอใกล้ถึงจุดหมาย จะเห็นการเคลื่อนที่แบบแยกทางคล้ายกับสถานการณ์ที่ 7

ผลของโปรแกรมที่สร้างเลียนแบบสถานการณ์จำลองที่ 10 ซึ่งได้รับรูปแบบของฝูงคนจากไฟล์ ซึ่งฝูงชนนี้เคลื่อนที่จากทางด้านซ้ายไปทางด้านขวา มีเซลเป้าหมาย 4 เซลคือ 20, 40, 60 และ 80 มีผลดังแสดงในรูปที่ 44

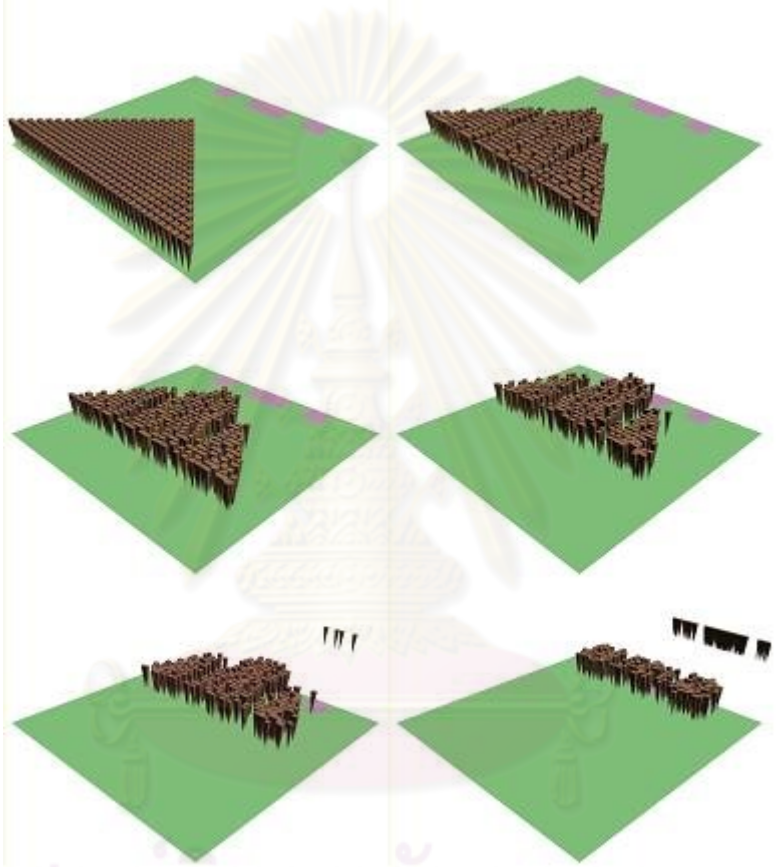


รูปที่ 44 แสดงภาพการเคลื่อนที่โดยรวมจากโปรแกรมในสถานการณ์ที่ 10

จากการสังเกตพฤติกรรม พบว่ามีการเคลื่อนไหวที่เป็นระเบียบคล้ายกับต้นฉบับ แต่พอใกล้ถึงจุดหมาย จะเห็นการเคลื่อนที่แบบแยกทางอย่างเห็นได้ตรงเซลที่ไม่ได้อยู่แนวเดียวกับเซลเป้าหมาย และเอเจนต์ที่อยู่ทางด้านข้างที่อยู่ห่างจากเป้าหมายจะพยายามเบียดเข้าสู่เป้าหมาย มีการรวมตัวกันของเอเจนต์เกิดขึ้นเนื่องจากเซลลูลาร์ฟลอกกิงไม่มีการตรวจจับการชน

4.4.2. พฤติกรรมที่ได้จากอัลกอริทึมฟลอกกิงแบบธรรมดา

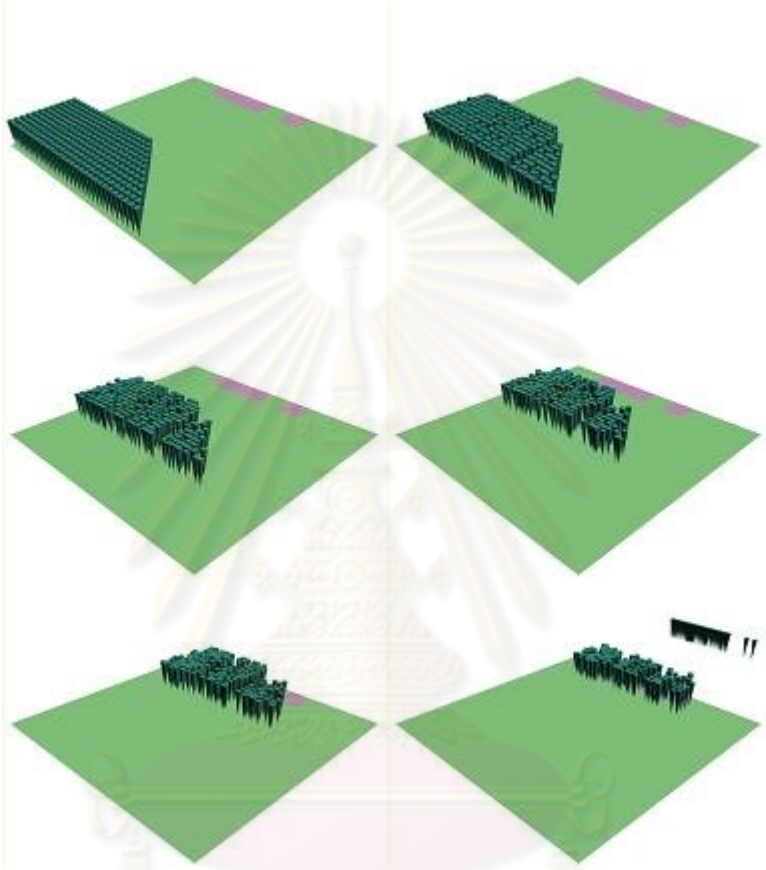
ผลของโปรแกรมที่สร้างเลียนแบบสถานการณ์จำลองที่ 1 ซึ่งได้รับรูปแบบของฝูงคนจากไฟล์ ซึ่งฝูงชนนี้เคลื่อนที่จากทางด้านซ้ายมือไปทางด้านขวามือ มีเซลเป้าหมาย 4 เซลคือหมายเลข 30 50 60 และ 80 มีผลดังแสดงในรูปที่ 45



รูปที่ 45 แสดงภาพการเคลื่อนที่โดยรวมจากโปรแกรมในสถานการณ์ที่ 1

จากการสังเกตพฤติกรรม พบว่ามีการเดินที่ค่อนข้างเกาะกลุ่มกันไปอย่างไม่เป็นระเบียบ และรูปร่างการเดินยังคงรูปแบบคล้ายกับสถานการณ์ต้นฉบับ แต่ก็สามารถเห็นเอเจนต์เดินแตกกลุ่มเพื่อไปยังเป้าหมายที่ใกล้กว่า เมื่อเอเจนต์เคลื่อนเข้าใกล้เป้าหมายดูเหมือนเอเจนต์จะหยุดรอก่อนเคลื่อนที่เข้าหาเป้าหมายเนื่องจากเป้าหมายเป็นเพียงจุดเพียงจุดเดียว วิธีป้องกันปัญหาเช่นนี้คือการขยายขอบเขตจุดเป้าหมายให้กว้างขึ้น ในส่วนของการชนนั้น เอเจนต์ไม่มีการรวมร่างกัน เพราะมีอัลกอริทึมป้องกันการชนอยู่

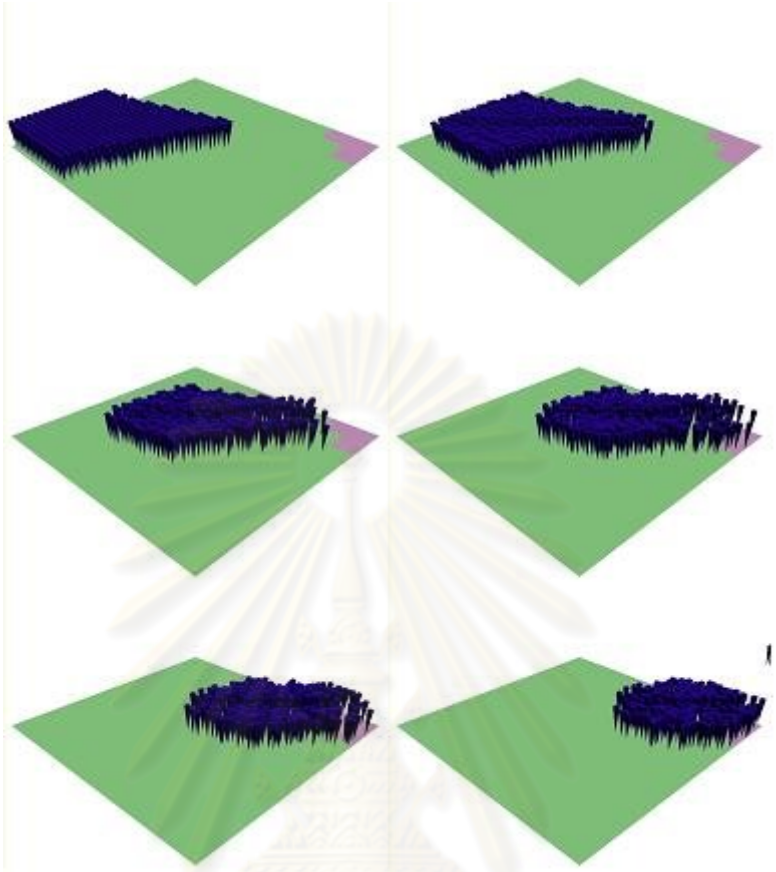
ผลของโปรแกรมที่สร้างเลียนแบบสถานการณ์จำลองที่ 2 ซึ่งได้รับรูปแบบของฝูงคนจากไฟล์ ซึ่งฝูงชนนี้เคลื่อนที่จากทางด้านซ้ายมือไปทางด้านขวามือ มีเซลเป้าหมาย 4 เซลคือ เซลหมายเลข 30 40 50 และ 70 มีผลดังแสดงในรูปที่ 46



รูปที่ 46 แสดงภาพการเคลื่อนที่โดยรวมจากโปรแกรมในสถานการณ์ที่ 2

จากการสังเกตพฤติกรรม พบว่ามีการเดินที่ค่อนข้างเกาะกลุ่มกันไปอย่างไม่เป็นระเบียบ จะเห็นการเดินแยกกลุ่ม ของกลุ่มเอเจนต์ที่อยู่ทางด้านขวาแล้วเคลื่อนที่เข้าสู่เป้าหมาย ส่วนเอเจนต์กลุ่มที่อยู่ทางด้านซ้ายเมื่อเคลื่อนที่เข้าใกล้เป้าหมายจะเกิดการเบียดกันเข้าหาเป้าหมาย แต่การรวมตัวกันนั้นไม่เกิดขึ้น

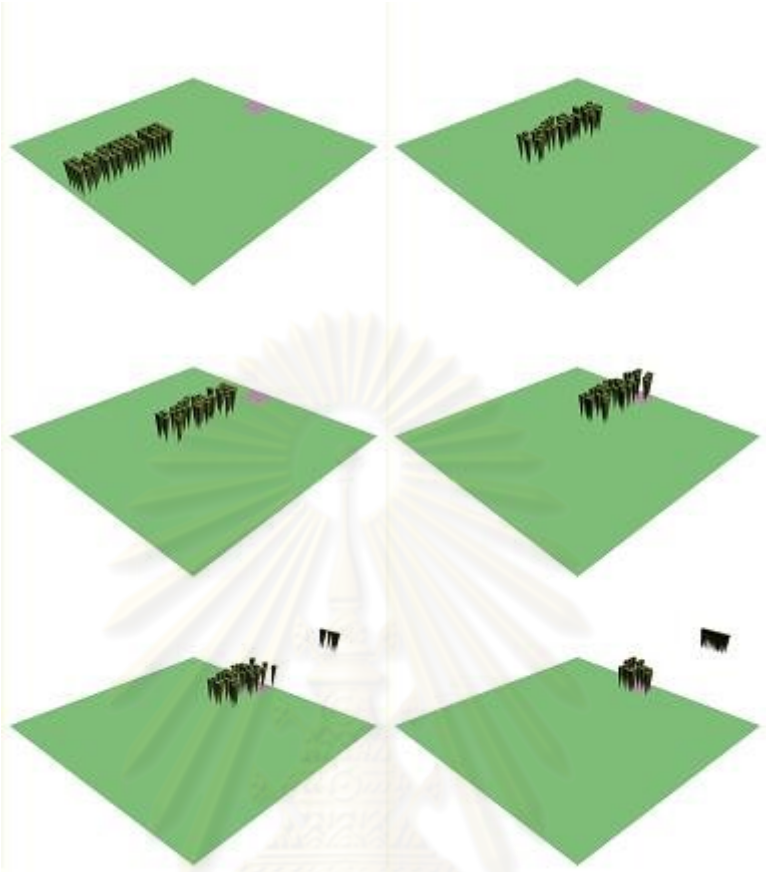
ผลของโปรแกรมที่สร้างเลียนแบบสถานการณ์จำลองที่ 3 ซึ่งได้ได้รับรูปแบบของฝูงคนจากไฟล์ ซึ่งฝูงชนนี้เคลื่อนที่จากทางด้านบนซ้ายไปทางด้านล่างขวา มีเซลเป้าหมาย 3 เซล คือ 90, 99 และ 100 มีผลดังแสดงในรูปที่ 47



รูปที่ 47 แสดงภาพการเคลื่อนที่โดยรวมจากโปรแกรมในสถานการณ์ที่ 3

จากการสังเกตพฤติกรรม พบว่ามีการเดินที่ค่อนข้างเกาะกลุ่มกันไปอย่างไม่เป็นระเบียบ และรูปร่างการเดินยังคงรูปแบบคล้ายกับสถานการณ์ต้นฉบับ เมื่อเอเจนต์เคลื่อนเข้าใกล้เป้าหมายเอเจนต์เคลื่อนแยกออกไปในเป้าหมายทั้ง 3 เป้าหมาย การแยกกลุ่มนั้นเกิดขึ้นเร็วกว่าเมื่อใช้เซลลูลาร์ฟลอกกิง ทั้งนี้อาจเนื่องมาจากการที่จุดเป้าหมายไม่ได้กำหนดเป็นพื้นที่ แต่กำหนดเป็นจุด

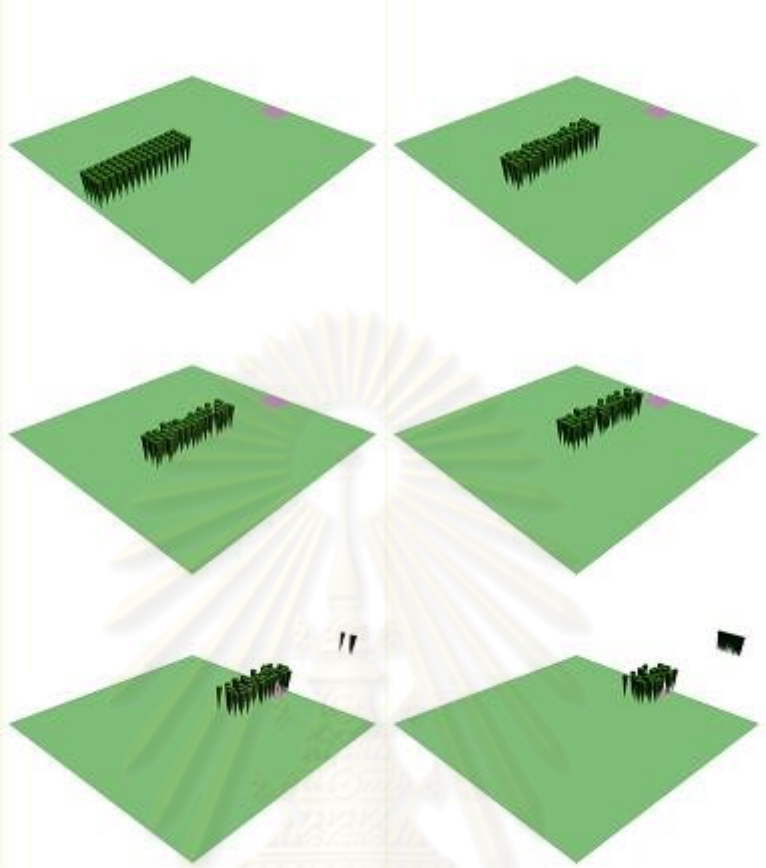
ผลของโปรแกรมที่สร้างเลียนแบบสถานการณ์จำลองที่ 4 ซึ่งได้รับรูปแบบของฝูงคนจากไฟล์ ซึ่งฝูงชนนี้เคลื่อนที่จากทางด้านซ้ายไปทางด้านขวา มีเซลเป้าหมาย 1 เซลคือเซลหมายเลข 50 มีผลดังแสดงในรูปที่ 48



รูปที่ 48 แสดงภาพการเคลื่อนที่โดยรวมจากโปรแกรมในสถานการณ์ที่ 4

จากการสังเกตพฤติกรรม พบว่ามีการเดินที่ค่อนข้างเกาะกลุ่มกันไปอย่างไม่เป็นระเบียบ ในระหว่างการเดินมีการเดินเบียดกันด้วย แต่มีการชนอยู่บ้าง ซึ่งการเดินแบบนี้เหมือนต้นฉบับมากกว่า และดูสมจริงมากกว่าเมื่อใช้เซลลูลาร์ฟลอกกิง

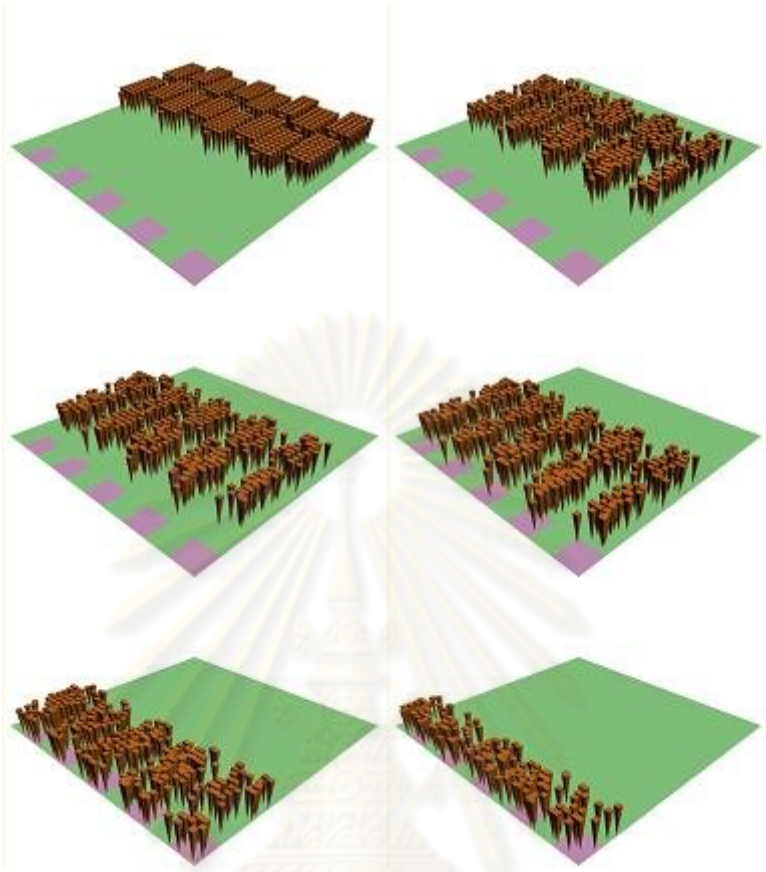
ผลของโปรแกรมที่สร้างเลียนแบบสถานการณ์จำลองที่ 5 ซึ่งได้รับรูปแบบของฝูงคนจากไฟล์ ซึ่งฝูงชนนี้เคลื่อนที่จากทางด้านซ้ายไปทางด้านขวา มีเซลเป้าหมาย 1 เซลคือเซลหมายเลข 60 มีผลดังแสดงในรูปที่ 49



รูปที่ 49 แสดงภาพการเคลื่อนที่โดยรวมจากโปรแกรมในสถานการณ์ที่ 5

จากการสังเกตพฤติกรรม พบว่ามีการเดินที่เกาะกลุ่มกันไปอย่างไม่เป็นระเบียบ ซึ่งในกรณีนี้จะเดินไม่เหมือนกับต้นฉบับเท่าไรนัก เพราะภาพยนตร์ต้นฉบับนั้นแสดงการเดินที่เป็นระเบียบ แสดงให้เห็นว่า ฟลอกกิงแบบธรรมดานั้น ใช้กับการเดินที่เป็นระเบียบไม่ดีนัก

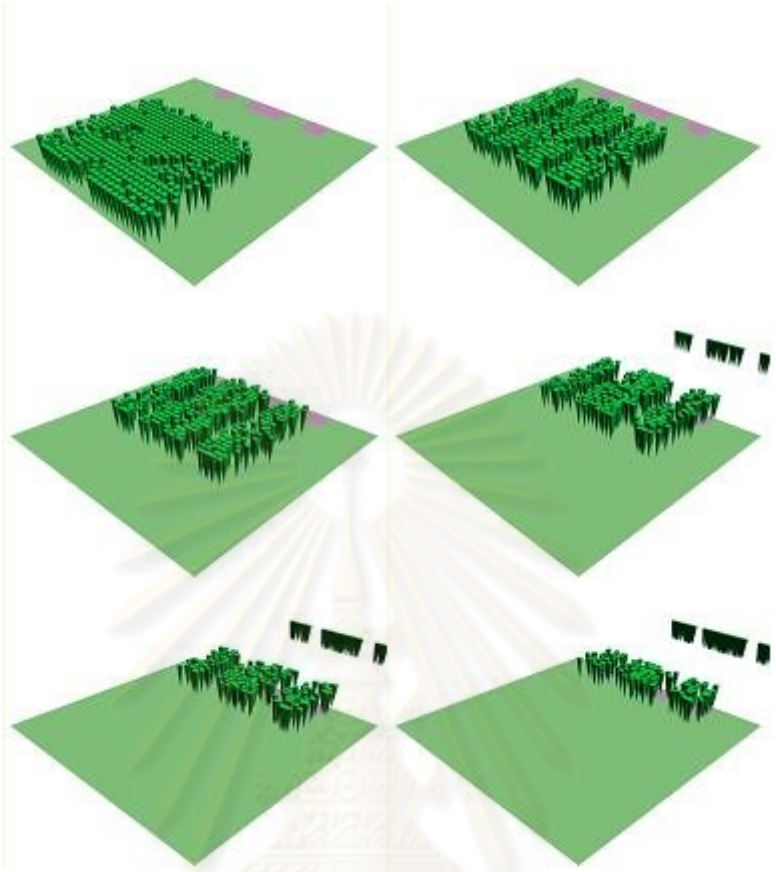
ผลของโปรแกรมที่สร้างเลียนแบบสถานการณ์จำลองที่ 6 ซึ่งได้รับรูปแบบของฝูงคนจากไฟล์ ซึ่งฝูงชนนี้เคลื่อนที่จากทางด้านขวาไปทางด้านซ้าย มีเซลเป้าหมาย 5 เซลคือ 11, 31, 51, 71 และ 91 มีผลดังแสดงในรูปที่ 50



รูปที่ 50 แสดงภาพการเคลื่อนที่โดยรวมจากโปรแกรมในสถานการณ์ที่ 6

จากการสังเกตพฤติกรรม พบว่าเอเจนต์ที่ไม่ได้อยู่ในแนวเดียวกับเป้าหมายนั้นพยายามเบียดเข้าหาเป้าหมายจนทำให้เกิดการรวมกลุ่มของเอเจนต์เพื่อเคลื่อนที่เข้าสู่เป้าหมาย ซึ่งเป็นพฤติกรรมที่คล้ายกับเมื่อใช้เซลลูลาร์ฟลอกกิง แต่ความเป็นระเบียบของแถวจะเริ่มเสียเมื่อเคลื่อนที่ไปได้ระยะหนึ่ง

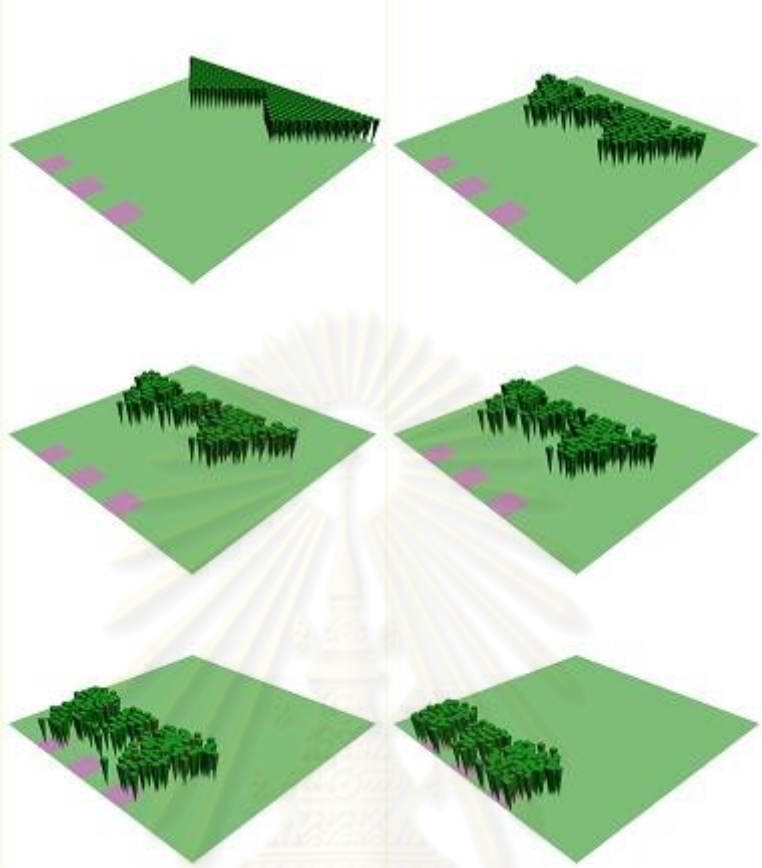
ผลของโปรแกรมที่สร้างเลียนแบบสถานการณ์จำลองที่ 7 ซึ่งได้รับรูปแบบของฝูงคนจากไฟล์ ซึ่งฝูงชนนี้เคลื่อนที่จากทางด้านขวาไปทางด้านซ้ายมีเซลเป้าหมาย 4 เซลคือ 30, 50, 60 และ 80 มีผลดังแสดงในรูปที่ 51



รูปที่ 51 แสดงภาพการเคลื่อนที่โดยรวมจากโปรแกรมในสถานการณ์ที่ 7

จากการสังเกตพฤติกรรม พบว่าเอเจนต์ที่ไม่ได้อยู่ในแนวเดียวกับเป้าหมายนั้น พยายามเบียดเข้าหาเป้าหมายอย่างชัดเจนจึงทำดูเหมือนการเข้าแถวเดินเข้าออกประตู การแยกกลุ่มออกจากกันเกิดขึ้นเร็วกว่าเมื่อใช้เซลลูลาร์ฟลอกกิง เช่นเดียวกับสถานการณ์ที่ 3

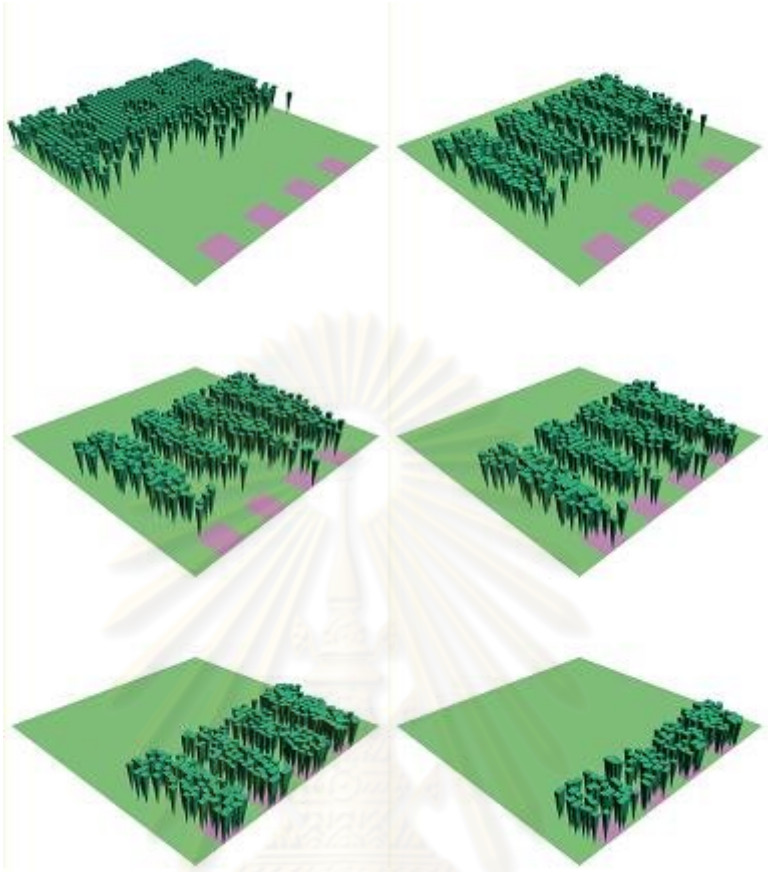
ผลของโปรแกรมที่สร้างเลียนแบบสถานการณ์จำลองที่ 8 ซึ่งได้รับรูปแบบของฝูงคนจากไฟล์ ซึ่งฝูงชนนี้เคลื่อนที่จากทางด้านขวาไปทางด้านซ้าย มีเซลเป้าหมาย 4 เซลคือ 11, 31, 51 และ 71 มีผลดังแสดงในรูปที่ 52



รูปที่ 52 แสดงภาพการเคลื่อนที่โดยรวมจากโปรแกรมในสถานการณ์ที่ 8

จากการสังเกตพฤติกรรม พบว่าเอเจนต์ที่ไม่ได้อยู่ในแนวเดียวกับเป้าหมายนั้นพยายามเบียดเข้าหาเป้าหมายจนทำให้เกินการรวมกลุ่มของเอเจนต์เพื่อเคลื่อนที่เข้าสู่เป้าหมาย การเคลื่อนที่โดยรวมยังคงรูปแบบคล้ายกับสถานการณ์ต้นแบบ แต่ลักษณะความเป็นระเบียบจะน้อยกว่าการใช้เซลลูลาร์ฟลอกกิงอย่างเห็นได้ชัด

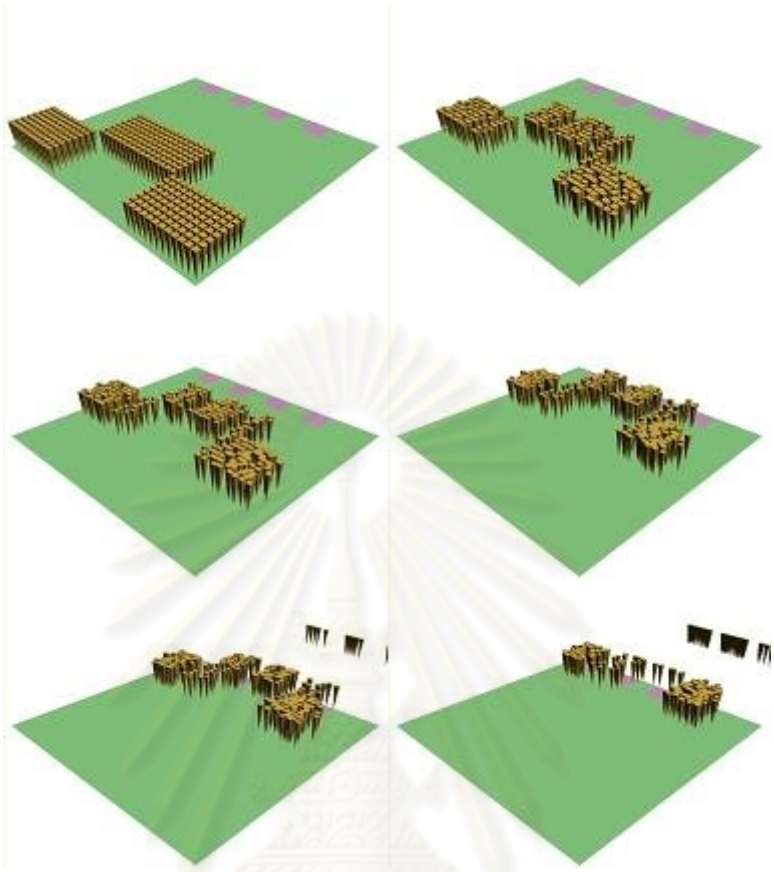
ผลของโปรแกรมที่สร้างเลียนแบบสถานการณ์จำลองที่ 9 ซึ่งได้รับรูปแบบของฝูงคนจากไฟล์ ซึ่งฝูงชนนี้เคลื่อนที่จากทางด้านบนไปทางด้านล่าง มีเซลเป้าหมาย 4 เซลคือ 92, 94, 96 และ 98 มีผลดังแสดงในรูปที่ 53



รูปที่ 53 แสดงภาพการเคลื่อนที่โดยรวมจากโปรแกรมในสถานการณ์ที่ 9

จากการสังเกตพฤติกรรม พบว่าเอเจนต์ที่ไม่ได้อยู่ในแนวเดียวกับเป้าหมายนั้นพยายามเบียดเข้าหาเป้าหมาย คล้ายกับสถานการณ์ที่ 7

ผลของโปรแกรมที่สร้างเลียนแบบสถานการณ์จำลองที่ 10 ซึ่งได้รับรูปแบบของฝูงคนจากไฟล์ ซึ่งฝูงชนนี้เคลื่อนที่จากทางด้านซ้ายไปทางด้านขวา มีเซลเป้าหมาย 4 เซลคือ 20, 40, 60 และ 80 มีผลดังแสดงในรูปที่ 54



รูปที่ 54 แสดงภาพการเคลื่อนที่โดยรวมจากโปรแกรมในสถานการณ์ที่ 10

จากการสังเกตพฤติกรรม พบว่าเอเจนต์เคลื่อนที่แบบแบ่งกลุ่มเพื่อเข้าหาเป้าหมาย และด้านข้างทั้ง 2 ฝั่งก็เคลื่อนที่เบียดเข้ามาทางด้านข้างเพื่อเข้าหาเป้าหมายเช่นกันแต่ไม่มีการชนให้เห็น การเกาะกลุ่มนั้นสูญเสียรูปแบบการตั้งแถวไปอย่างรวดเร็ว แต่ยังคงลักษณะกลุ่มสามกลุ่มตามต้นฉบับ

4.5 ผลการใช้ทรัพยากรของคอมพิวเตอร์

รายละเอียดของเครื่องคอมพิวเตอร์ที่ใช้ในการประมวลผลหน่วยประมวลผลกลางหน่อยประมวลผลทางกราฟิก หน่วยความจำหลัก ขนาดสื่อบันทึกข้อมูล

CPU Intel® Core(TM) 2 Quad Q9450 @ 2.66GHz

RAM 4 GB

Hard disk 320 GB

GPU ATI Radeon HD 4800 Series

ผลการใช้ทรัพยากรทางคอมพิวเตอร์ของโปรแกรมเซลลูลาร์ฟลอกกิง กับโปรแกรมมัลติเอเจนต์ฟลอกกิง ซึ่งการวัดผลนั้น วัดจากการใช้ หน่วยประมวลผลกลางหรือ CPU และการใช้หน่วยความจำหรือ RAM เพื่อดูว่ามีกรณีที่เป็นข้อยกเว้นหรือไม่ นอกจากนี้ยังวัดเวลาที่ใช้ในการประมวลผลโดยนับตั้งแต่ เวลาที่โปรแกรมเริ่มประมวลผลจนถึงเวลาที่โปรแกรมประมวลผลถึงเฟรมที่ 400 หรือ โปรแกรมทำงานครบ 400 รอบ ซึ่งผลที่ได้จากการวัดของโปรแกรมเซลลูลาร์ฟลอกกิง กับ โปรแกรมมัลติเอเจนต์ฟลอกกิงนั้น แสดงในตารางที่ 1 และตารางที่ 2 ตามลำดับ

ตารางที่ 1 แสดงการใช้ทรัพยากรของเครื่องของโปรแกรมเซลลูลาร์ฟลอกกิง

การใช้ทรัพยากรของเซลลูลาร์ฟลอกกิง					
สถานการณ์ที่	จำนวนเอเจนต์	เซลเป้าหมาย	การใช้งานโพรเซสเซอร์ (%)	การใช้งานหน่วยความจำ (MB)	เวลาที่ใช้ในการคำนวณการเคลื่อนที่(นาที)
1	400	30, 50, 60, 80	25	348.310	0.08
2	306	30, 40, 50, 70	25	364.439	0.07
3	379	90, 99, 100	25	372.729	0.09
4	52	50	25	367.478	0.02
5	64	60	25	355.840	0.02
6	529	11, 31, 51, 71, 91	25	368.784	0.11
7	406	30, 50, 60, 80	25	363.403	0.08
8	287	21, 41, 61	25	360.520	0.07

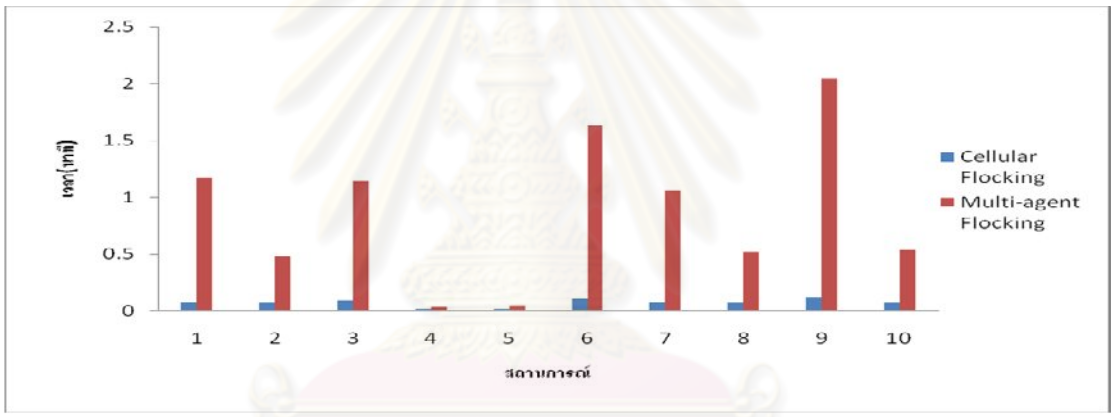
การใช้ทรัพยากรของเซลลูลาร์ฟลอกกิง					
สถานการณ์ที่	จำนวนเอเจนต์	เซลเป้าหมาย	การใช้งานโพรเซสเซอร์ (%)	การใช้งานหน่วยความจำ (MB)	เวลาที่ใช้ในการคำนวณการเคลื่อนที่(นาที)
9	562	92, 94, 96, 98	25	376.182	0.12
10	320	20, 40, 60, 80	25	364.524	0.07

ตารางที่ 2 แสดงการใช้ทรัพยากรของเครื่องของโปรแกรมมัลติเอเจนต์ฟลอกกิง

การใช้ทรัพยากรของโปรแกรมมัลติเอเจนต์ฟลอกกิง					
สถานการณ์ที่	จำนวนเอเจนต์	เป้าหมาย	การใช้งานโพรเซสเซอร์ (%)	การใช้งานหน่วยความจำ (MB)	เวลาที่ใช้ในการคำนวณการเคลื่อนที่(นาที.)
1	400	30, 50, 60, 80	25	352.422	1.17
2	306	30, 40, 50, 70	25	359.209	0.48
3	379	90, 99, 100	25	372.948	1.14
4	52	50	25	356.130	0.03
5	64	60	25	347.029	0.05
6	529	11, 31, 51, 71, 91	25	377.411	1.63
7	406	30, 50, 60, 80	25	375.493	1.06
8	287	21, 41, 61	25	362.007	0.52
9	562	92, 94, 96, 98	25	385.164	2.05

การใช้ทรัพยากรของโปรแกรมมัลติเอเจนต์ฟลอกกิง					
สถานการณ์ที่	จำนวนเอเจนต์	เป้าหมาย	การใช้งานโพรเซสเซอร์ (%)	การใช้งานหน่วยความจำ (MB)	เวลาที่ใช้ในการคำนวณการเคลื่อนที่(นาที.)
10	320	20, 40, 60, 80	25	379.493	0.54

จากข้อมูลในตารางที่ 1 กับ 2 เมื่อนำข้อมูลของแต่ละสถานการณ์จำลองมาเปรียบทางด้านเวลาที่ใช้ในการประมวลผลจะเห็นความแตกต่างของเวลาที่ใช้ในการประมวลผลอย่างชัดเจนดังแสดงในรูปที่ 55



รูปที่ 55 แสดงผลการเปรียบเทียบเวลาที่ใช้ในการประมวลผลของเซลลูลาร์ฟลอกกิงกับมัลติเอเจนต์ฟลอกกิง

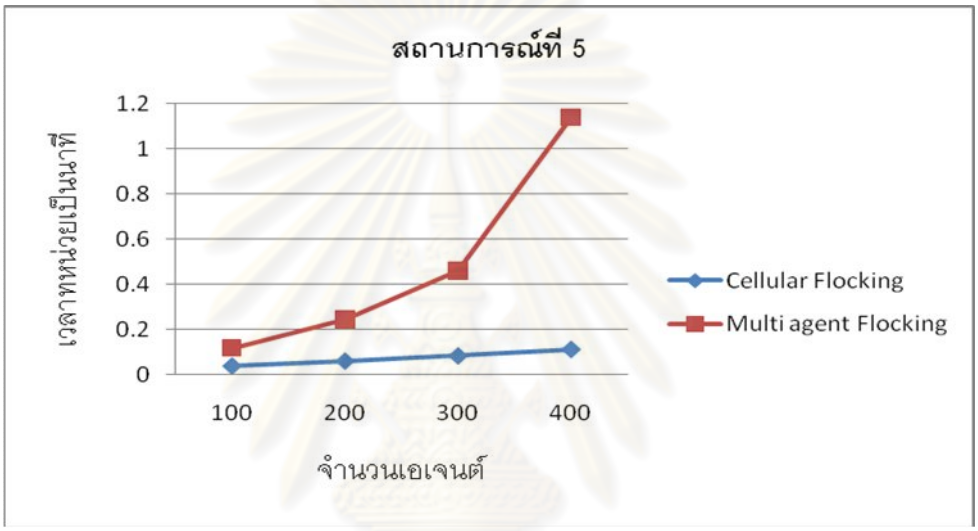
ซึ่งการเปรียบเทียบเวลาและการใช้หน่วยความจำระหว่างเซลลูลาร์ฟลอกกิงกับมัลติเอเจนต์ฟลอกกิงอย่างละเอียดนั้น แสดงในตารางที่ 3

ตารางที่ 3 แสดงการเปรียบการใช้ทรัพยากรของทั้ง 2 โปรแกรม

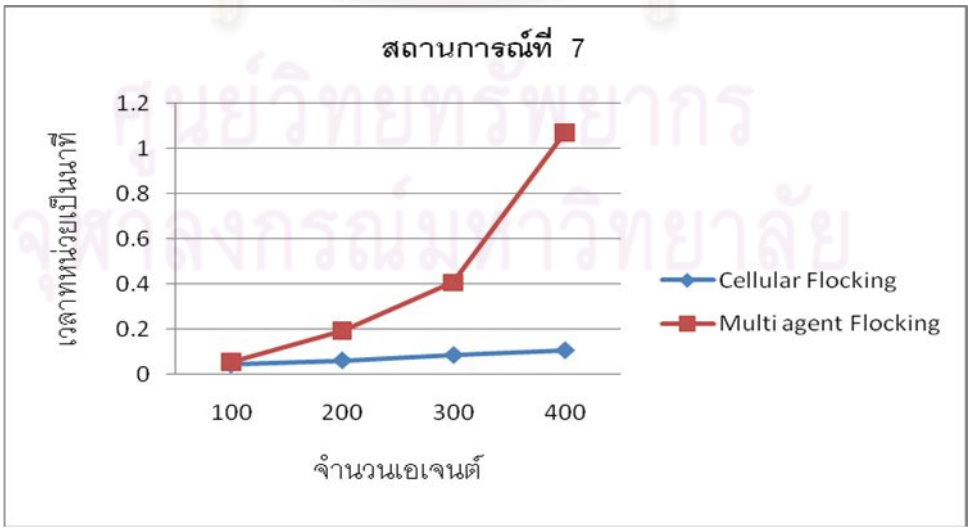
	เซลลูลาร์ฟลอกกิง			มัลติเอเจนต์ฟลอกกิง		
สถานการณ์ที่	การใช้งานโพรเซสเซอร์ (%)	การใช้งานหน่วยความจำ(MB)	เวลาที่ใช้ในการคำนวณการเคลื่อนที่ (นาที.)	การใช้งานโพรเซสเซอร์ (%)	การใช้งานหน่วยความจำ (MB)	เวลาที่ใช้ในการคำนวณการเคลื่อนที่ (นาที.))
1	25	348.310	0.08	25	352.422	1.17
2	25	364.439	0.07	25	359.209	048
3	25	372.729	0.09	25	372.948	1.14
4	25	367.478	0.02	25	356.130	0.03
5	25	355.840	0.02	25	347.029	0.05
6	25	368.784	0.11	25	377.411	1.63
7	25	363.403	0.08	25	375.493	1.06
8	25	360.520	0.07	25	362.007	0.52
9	25	376.182	0.12	25	385.164	2.05
10	25	364.524	0.07	25	379.493	0.54

ผลการใช้ทรัพยากรทางโพรเซสเซอร์และหน่วยความจำของฟลอกกิงทั้งสองแบบนั้นมีความใกล้เคียงกัน จึงแน่ใจได้ว่าไม่มีกรณียกเว้นเกิดขึ้น ส่วนเวลาในการคำนวณเพื่อแสดงอนิเมชันนั้น เซลลูลาร์ฟลอกกิงใช้ต่ำกว่ามาก ยิ่งจำนวนเอเจนต์มีมาก ความแตกต่างยิ่งเห็นได้ชัด จากผลการทดลองโดยเฉลี่ยแล้วจะใช้เวลาในการรันดีกว่าประมาณ 11.87เท่า มีเพียงสถานการณ์ที่ 4 และ 5 เท่านั้นที่ใช้เวลาใกล้เคียงกัน ทั้งนี้เพราะจำนวนเอเจนต์ในสถานการณ์เหล่านี้มีน้อย

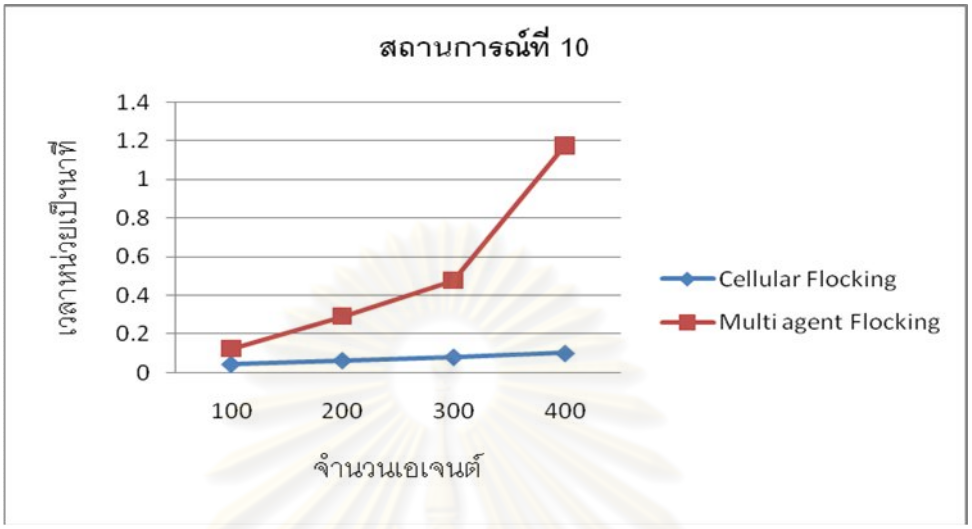
จากสถานการณ์จำลองทั้ง 10 แบบข้างต้นเราได้เลือกสถานการณ์ที่ 5, 7 และ 10 มาเป็นตัวแทนของการเดินของฝูงชนกลุ่มใหญ่ในลักษณะต่างกัน (เป็นแถว เป็นกลุ่มกระจาย และเป็นกลุ่มย่อยๆหลายกลุ่ม) มาทำการทดลองเพิ่มเติมโดยกำหนดให้แผนที่จุเอเจนต์ 100 ตัว 200 ตัว 300 ตัว และ 400 ตัวตามลำดับ โดยทำการเปรียบเทียบเวลาที่ใช้ในการประมวลผลสำหรับการทำงานของเซลลูลาร์ฟลอกกิงและมัลติเอเจนต์ฟลอกกิงซึ่งผลการเปรียบเทียบเวลาของสถานการณ์ที่ 5, 7และ 10 แสดงอยู่ในรูปที่ 56, 57 และ 58 ตามลำดับ



รูปที่ 56 แสดงการเปรียบเทียบการใช้เวลาของเอเจนต์จำนวน 100, 200, 300 และ 400 ตัว ในสถานการณ์ที่ 5



รูปที่ 57 แสดงการเปรียบเทียบการใช้เวลาของเอเจนต์จำนวน 100, 200, 300 และ 400 ตัวในสถานการณ์ที่ 7



รูปที่ 58 แสดงการเปรียบเทียบการใช้เวลาของเอเจนต์จำนวน 100, 200, 300 และ 400 ตัว ในสถานการณ์ที่ 10

จากกราฟจะเห็นว่าเวลาที่เซลลูลาร์ฟลอกกิงใช้นั้นไม่เพิ่มขึ้นเท่าใดแม้ว่าจำนวนเอเจนต์จะเพิ่มขึ้นก็ตาม ส่วนมัลติเอเจนต์นั้นเวลาที่ใช้ขึ้นอยู่กับจำนวนของเอเจนต์เป็นอย่างมาก ยิ่งจำนวนของเอเจนต์มาก มัลติเอเจนต์นั้นจะใช้เวลาเป็นเอกซ์โปเนนเชียล

ตารางที่ 4 แสดงผลคะแนนจากการทำแบบสอบถาม ซึ่งความเหมือนต้นฉบับนั้น มัลติเอเจนต์ฟลอกกิงนั้นเหมือนมากกว่า มีสถานการณ์ที่ 5 เพียงสถานการณ์เดียวที่คะแนนของเซลลูลาร์ฟลอกกิงมีค่าเท่ากับมัลติเอเจนต์ฟลอกกิง นอกนั้นกลุ่มตัวอย่างให้คะแนนผลของมัลติเอเจนต์มากกว่าทุกสถานการณ์ แต่เซลลูลาร์ฟลอกกิงนั้นแสดงความเหมือนในระดับที่ผู้สังเกตกลุ่มตัวอย่างยอมรับได้จากการสังเกตการเคลื่อนที่ในระยะจำกัด (400 ก้าว) โดยมีคะแนนเฉลี่ยจาก 10 เท่ากับ 7.11 ในขณะที่มัลติเอเจนต์ฟลอกกิงนั้นมีคะแนนความเหมือนต้นฉบับโดยเฉลี่ยตามความเห็นของผู้ทดลองเท่ากับ 7.56

ตารางที่ 4 แสดงผลคะแนนจากการทำแบบสอบถาม

สถานการณ์ที่	คะแนนเซลลูลาร์ฟลอกกิง	คะแนนมัลติเอเจนต์ฟลอกกิง
1	7.1	7.7
2	7	7.6
3	7	7.6
4	7.5	7.6
5	7.6	7.6
6	6.9	7.4
7	7.2	7.3
8	6.9	7.4
9	7.4	7.8
10	6.5	7.6

บทที่ 5

สรุปผลการทดลอง อภิปรายผล และข้อเสนอแนะ

5.1 สรุปผลการทดลอง และอภิปราย

งานวิทยานิพนธ์นี้นำเสนอการทำฟลอกกิงในระดับเซล เพื่อลดการคำนวณของการสร้างอนิเมชันสำหรับฝูงชน ผลการทดลองแสดงให้เห็นว่า การทำฟลอกกิงในระดับเซลนั้นช่วยลดการคำนวณลงได้เป็นอย่างมาก ซึ่งจากตัวอย่างที่ทดลอง สามารถลดเวลาการคำนวณโดยรวมได้ถึง11.87 เท่าแต่เอเจนต์ต้องสูญเสียลักษณะพฤติกรรมส่วนตัวไป

จากแบบสอบถาม สถานการณ์ที่ 1, 2 , 3, 4, 6, 7, 8, 9 และ 10ผลที่ได้จากโปรแกรมมัลติเอเจนต์ฟลอกกิง มีความคล้ายกับสถานการณ์จำลองต้นแบบมากกว่าผลที่ได้จากโปรแกรมเซลลูลาร์ฟลอกกิง เนื่องจากการเคลื่อนที่ของกลุ่มเอเจนต์นั้นมีความไม่เป็นระเบียบอยู่ในตัว ซึ่งเซลลูลาร์ฟลอกกิงไม่สามารถทำการเคลื่อนที่ที่ไม่เป็นระเบียบหรือการเคลื่อนที่ที่ไม่เหมือนกันของแต่ละเอเจนต์ได้ดี นอกจากนี้ เมื่อมีสถานการณ์ที่ทำให้เกิดการชน เช่น ทางออกมีลักษณะแคบ เอเจนต์ในเซลลูลาร์ฟลอกกิงจะเบียดเสียดจนรวมตัวกันเพราะไม่มีการป้องกันการชน ทำให้ความเหมือนสถานการณ์ต้นฉบับลดลงไป

สถานการณ์ที่ 5 เป็นเพียงสถานการณ์เดียวที่ผู้ตอบแบบสอบถามให้คะแนนเซลลูลาร์ฟลอกกิงเท่ากันกับมัลติเอเจนต์ฟลอกกิง ทั้งนี้เพราะฝูงชนในสถานการณ์ที่ 5 นั้นมีการเคลื่อนที่ที่เป็นระเบียบอย่างมาก

แม้จะสูญเสียพฤติกรรมส่วนตัวไป แต่พฤติกรรมของเซลลูลาร์ฟลอกกิงนั้น ยังถือว่าเป็นที่ยอมรับได้ โดยมีคะแนนเฉลี่ยอยู่ที่ 7.11 จาก 10 จากแบบสอบถาม

ในการนำไปใช้นั้น เซลลูลาร์ฟลอกกิงน่าจะเหมาะกับการทำฝูงชนที่เดินแถวอย่างเป็นกลุ่มก้อนเป็นส่วนใหญ่ เช่นการเดินทัพของทหารซึ่งเอเจนต์ส่วนใหญ่จะเคลื่อนไหวเป็นกลุ่มก้อนแต่ยังคงอนุญาตให้เอเจนต์ที่อยู่นอกแถวเคลื่อนที่ร่วมกับกลุ่มเอเจนต์ส่วนใหญ่ได้ ทำให้การเคลื่อนไหวไม่เป็นรูปแบบตายตัวจนเกินไป นอกจากนี้ยังสามารถใช้กับการจำลองฝูงชนในระยะไกล เพื่อไม่ให้การชนเป็นที่สังเกตได้ง่าย ถ้าจำนวนคนน้อยผลของเซลลูลาร์ฟลอกกิงเห็นได้ชัดว่ามีการเรียงตัวมากเกินไป ส่วนถ้าจำนวนคนมากผลของเซลลูลาร์ฟลอกกิงจะเห็นได้ไม่ชัดเจนพฤติกรรมที่ยอมรับได้ ซึ่งได้จากการทดลองนั้น ได้จากระยะการเคลื่อนที่ซึ่งยังถือว่าค่อนข้างจำกัด ซึ่งทำให้เซลลูลาร์ฟลอกกิงเหมาะกับการจำลองกลุ่มคนที่มีจำนวนมากในระยะการเคลื่อนที่จำกัด

5.2 ข้อเสนอแนะ

5.2.1 การปรับปรุงในส่วนของเซลลูลาร์ฟลอกกิง

สิ่งที่เป็นปัญหาสำคัญคือ ความเป็นระเบียบจนเกินไป กับการชนและการเดินกลับไปกลับมาของเอเจนต์บางตัว การแก้ปัญหาความเป็นระเบียบจนเกินไปนั้น อาจใช้การสุ่มขนาดของความเร็วให้ต่างกันสำหรับเอเจนต์แต่ละตัว โดยอาศัยความเร็วที่คำนวณในระดับเซลเป็นฐาน แต่ก็จะใช้การคำนวณที่มากขึ้นอย่างเห็นได้ชัด เพราะต้องคำนวณความเร็วของเอเจนต์แต่ละตัว

ส่วนการแก้ปัญหาการชนนั้น การแก้ปัญหาที่แน่นอนคือ การใช้การตรวจจับการชนในระดับเอเจนต์ แต่ก็จะทำให้การคำนวณช้าลงอย่างมาก อีกวิธีหนึ่งในการแก้ปัญหาการชนคือการกำหนดให้แต่ละเซลไม่สามารถจุเอเจนต์ได้มากในตอนเริ่มรัน ซึ่งโอกาสที่เอเจนต์จะชนกันนั้นก็จะลดลงไปด้วย

การเดินกลับไปกลับมาเกิดขึ้นเมื่อมีการเปลี่ยนเซลเป้าหมายของเอเจนต์ ซึ่งอาจใช้การจำกัดมุมของเซลมาแก้ไขได้

5.2.2 การปรับปรุงในส่วนของมัลติเอเจนต์ฟลอกกิง

ในส่วนของมัลติเอเจนต์นั้นควรจะมีการกำหนดเป้าหมายให้เอเจนต์แต่ละตัวเลยเมื่อต้องการสร้างการเคลื่อนที่แบบเป็นระเบียบ เนื่องจากในการทดลองนี้เราได้กำหนดจุดเป้าของมัลติเอเจนต์ฟลอกกิงไว้เพียงจุดๆ เดียวซึ่งอาจจะทำให้ทิศทางการเคลื่อนที่ของเอเจนต์มีการเบียดเสียดกันมากเมื่อยิ่งเข้าใกล้เป้าหมาย อาจเป็นพราะจุดเป้าหมายมีขนาดเล็กเกินไป ซึ่งไม่เหมือนกับเซลลูลาร์ฟลอกกิงซึ่งคำนวณจุดหมายจากพื้นที่เซลทั้งเซล

รายการอ้างอิง

[1] Wikipedia. The Lion, the Witch and the Wardrobe [Online]. Available from: http://en.wikipedia.org/wiki/The_Lion,_the_Witch_and_the_Wardrobe [2010, 4 May]

[2] กันตนา แอนนิเมชั่น. ก้านกล้วย [Online]. Available from: http://www.kantana.com/khankluay/ [2010, 4 May]

[3] Mathworld. Cellular Automaton [Online]. Available from: http://mathworld.wolfram.com/CellularAutomaton.html [2008, 7 Feb]

[4] Andrew, I. Artificial War Multi agent-Based Simulation of Combat. Singapore: World Scientific, 2004.

[5] Reynolds.. Boids [Online].ปี, Available from : http://www.red3d.com/cwr/boids/ [2008, 3 mar]

[6] Wikipedia. Batman Returns - Wikipedia [Online]. Available from: http://en.wikipedia.org/wiki/Batman_Returns [2010, 28 Apr]

[7] Youtube. The Lion king [Online]. Available from: http://www.youtube.com/watch?v=ZKDw3qxR7dk [2008, 30 Mar]

[8] Reynolds, C.W. Flocks, Herds, and Schools: A Distributed Behavioral Model. ACM SIGGRAPH'87. pp.25-34. 1987.

[9] Massive. Massivesoftware [Online]. Available from: http://www.massivesoftware.com/whatismassive/ [2010, 28 Apr]

[10] Tantisiriwat, W., Sumleeon, S., and Kanongchaiyos, P. A Crowd Simulation Using Individual-Knowledge-Merge based path Construction and Smoothed Particle Hydrodynamics. The 15th International Conference in Central Europe on Computer Graphics, Visualization and Computer Vision'2007, Czech Republic. 2007.

[11] Saber, R.O., and Murry, R.M. Flocking with obstacle avoidance: cooperation with limited communication in mobile networks. Decision and Control, 2003. Proceedings.42nd IEEE Conference, pp.2022-2028. 2003.

[12] Leo, B., and Richard, H. Coherent Formation for Agents Using Flocking with Cellular Automata. In Knowledge-Based Intelligent Information and Engineering Systems. pp.220-227. SpringerLink, 2006.

[13] Masaru, A., and Akira, M. Massive Multi-Agent Simulation in 3D. In Soft Computing as Transdisciplinary Science and Technology. pp.295-305. 2005.

[14] Hamagami, T., and Hirata, H. Method of crowd simulation by using multi agent on cellualar automata. Intelligent Agent Technology, 2003.IAT 2003. IEEE/WIC international Conference. pp.46-52. 2003.

[15] Bandini, S., Manzoni, S., and Vizzari, G. Situated Cellular Agents for Crowd Simulation and Visualization [Online]. Available from : http://www.iemss.org/iemss2004.pdf/complexinteract/bandsitu.pdf [2009, 15 mar]

[16] Klupfel,H., and Meyer-Konig, T. Simulation of the Evacuation of a football stadium using the CA Model PedGo. In Traffic and Granular Flow'03. pp.423-428. SpringerLonk, 2007.

[17] Sweeter, P., and Wiles, J. Combining Influence Maps and cellular Automata for Reactive Game Agent. 6th International Conference on Intelligent Data Engineering and Automated Learning (IDEAL2005) july 2005.

[18] Murakami, Y., Minami, K., Kawasoe, T., and Ishida, T. Multi-Agent Simulation for Crisis Management. Knowledge Media Networking, 2002. Proceedings.IEEE Workshop. pp.135-139, 2002.

[19] Treuile, A., Cooper, S., and Popavic, Z. Continuum Crowds. SIGGRAPH '06 ACM, 2006

[20] Conrad Parker. Boids Pseudo code [Online]. Available from: http://www.vergenet.net/~conrad/boids/pseudocode.html [2010, 28 Apr]

ภาคผนวก

ภาคผนวก ก ผลงานการตีพิมพ์

Jongsarikit Kitikun and Kotrajaras Vishnu.2009. Cellular Flocking: Improving the Performance of Agent Based Crowds. CGAT'09 Annual International Conference and Industry Symposium on Computer Game: Animation, Multimedia, IPTV, Edutainment & IT Security. Singapore.

Cellular Flocking: Improving the performance of Agent Based Crowds

Kitikun Jongsarikit
Department of Computer Engineering
Faculty of Engineering
Chulalongkorn University
Payathai Road, Patumwan
Bangkok 10330, Thailand
Tel. +66817797655
kitikun_j@hotmail.com

Vishnu Kotrajaras
Department of Computer Engineering
Faculty of Engineering
Chulalongkorn University
Payathai Road, Patumwan
Bangkok 10330, Thailand
Tel. +66890212323
vishnu@cp.eng.chula.ac.th,
ajarntoe@gmail.com

Abstract

Commercial animation software utilizes its crowd feature based on agent technologies. Using an intelligent agent for one character allows animators to easily modify a specific character's behavior in detail, while most other characters can still use the same behavioral template. An agent based crowd, however, suffers from poor performance because a CPU needs to calculate each and every agent's decision. This paper presents an approach for reducing the CPU load. By giving agents in the same map cell a shared brain, a group decision can be made using flocking algorithm at cellular automata level. This reduces the calculations significantly. Maintaining the distance among agents and computing agents' direction are made into group decisions, while collision avoidance remains individuals. Our results show that the proposed technique not only reduces the calculations, it also maintains satisfactory individual-like movement for each agent.

Keywords

crowd animation, intelligent agent, flocking, cellular automata

1. Introduction

In recent years, there has been a great increase in crowd animation usage, especially in the entertainment business. Crowd animation has been used to produce bands of soldiers, flocks of birds, or even schools of fish. Such effect could be produced by using mathematical equations or by simulating individual agents. Mathematical based methodology was fast. However, software used in the film industry tended to utilize agent based crowd animation because it allowed artists to control agents intuitively. An agent in agent based crowd animation software had its own set of sensors, such as vision and hearing. To move agents to their common destination, an artist only needed to make an agent follow some noise or some kind of visual indicators (which would be generated from the destination). Each agent then calculated its own direction and speed towards the destination for that moment in time. An agent was able to navigate around its environment and other agents using rules given by

designers. Flocking rules were applied for group movements. Once an agent was defined, all agents of the same type behave in the same way. A variety in the crowd could be made by utilizing different motion capture or animation data.

Despite its intuitive interface, agent based crowd animation was very slow to render because every agent had to make its decision, resulting in a very intensive computation and longer production time. In this paper we propose Cellular Flocking, a methodology for reducing the number of individual flocking decisions. By putting agents into groups and treating each group like an agent, common direction for each group can be found and distances between groups can also be maintained. These results can approximate the usage of the flocking algorithm on each individual agent. This reduces the computation time for flocking considerably. Individual movements can still be displayed by adding randomness to selected characters.

2. Related Work

Tantisiriwat et al. [1] proposed a methodology where agents combined knowledge to construct their paths. Fluid equation was used to simulate crowd. However, a complete paths construction could not be achieved in a single run. Individuals could not be easily modified by scene designers because the equation was not easily understood by artists. In our work, paths were constructed using influence maps. Our crowd was constructed using agents, which were easier to work with.

Aoyagi and Namatame [2] extended Reynolds' flocking algorithm by modifying the rules of flocking. Individual agents were easier to be distinguished from each other. However, no group decisions were made in order to reduce the amount of calculation. Saber and Murray [3] presented a technique for obstacle avoidance in flocking. Agents were represented by nodes in a graph. Distances between agents were represented by edges. Nodes that were situated close to an obstacle kept a certain distance from it. Other nodes adjusted their positions and their distributions according to the nodes near the obstacle. This technique still did not cover shared decisions.

Leo Bi and Hall [4] combined cellular automata and flocking Algorithm. A cell in cellular automata was used to calculate a position for an agent. An agent's movement relied on its leader. Agents positioned themselves using a tree structure, with the leader as its root. Obstacle avoidance could be achieved by changing the shape of the tree. This technique utilized cellular automata, but using one cell per agent could not reduce the amount of calculation as well as it should. Hamagami and Hirata [5] simulated a crowd using a two layer model composed of cellular automata and multi-agent architecture. Cellular automata were used purely for calculation, while multi-agent was used to display agents' movement. One cell represented one agent; therefore shared decisions amongst agents were not utilized. Bandini

et al. [6] proposed a multi-agent system for crowd modeling based on situated cellular agents. One agent was represented by one cell but each agent could occupy a varied size of area. Agents interacted with environments through its sensors and information in cellular automata, which was similar to our work. Nevertheless, our shared decision amongst agents allowed better calculation optimization.

There existed works which allowed one cell to contain more than one agent. Sweetser [7] proposed EmerGEnT system, a system for creating units' AI in real-time strategy games. EmerGEnT used cellular automata for its map. Each cell of the map contained its environmental properties, which could influence agents' movement decisions. Although a cell could contain more than one agent, each of Sweetser's agents obtained information from neighboring cells and made its own movement decisions individually. On the contrary, our model had a cell read information from neighboring cells and made decision for all agents in the cell. In Sweetser's model, agents only obtained information from the map, not from other agents. Therefore collision avoidance between agents was not realized. Our model took care of this issue. Treuille [8] presented a real-time crowd model based on continuum dynamics. A dynamic potential function guided all individuals simultaneously without any need for explicit collision avoidance. A grid discretization scheme allowed the potential function to be applied on a cellular automata map to reduce the amount of calculation. Each character's speed was influenced by population density. This approach only modeled the potential function at cell level. It did not model individual agents. Therefore characters that shared the same grid were able to intersect. It was fixed by enforcing a minimum distance over all characters, but this still introduced artifacts. Not modeling individual agents, the system lost the flexibility and variability available in agent-based approaches. Our model also calculated most crowd movements from the cell level. However, we used agents to calculate movements within cells. This allowed better individual adjustments and improved behavior in smaller crowds.

3. Our Methodology

In order to implement our Cellular Flocking for a specified number of agents, a map had to be divided into rows and columns. Users could specify the number of rows and columns. The more rows or columns, the smaller each map cell would be. This would also result in smaller agent groups. In our prototype, agents were randomly distributed all over the map. Agents were distributed such that their original positions were not too close or overlapped. Once a destination was put on the map, a value that represented proximity to the destination was propagated from the destination cell to its neighbor cells and from its neighbor cells to their neighbors. No cell received this value twice. Agents that occupied the same cell were put into the same group. From its and its neighbors' information, a cell worked out a common speed and direction for its agents. The speed of an agent could change depending on the distance from other agents. We used a position database to store each agent's position. This allowed

us to quickly check whether a position (x, y) had which agent inside. Figure 1 illustrates our system.

The information needed to calculate the common speed and direction for a cell included:

- Proximity value from the cell and its neighbor cells to the destination.
- Common directions of the cell and its neighbor cells.
- Population density within the cell and its neighbor cells.

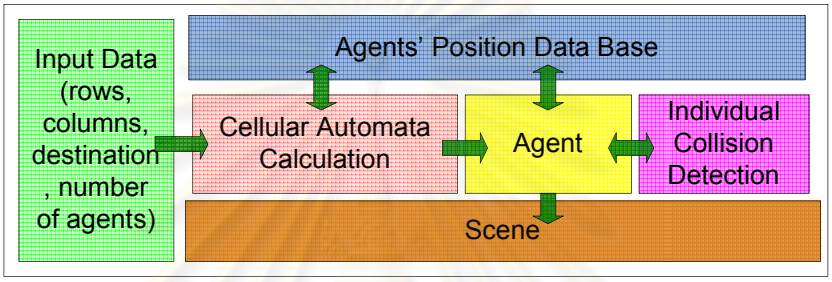


Figure 1. The Cellular Flocking Process

A cell had its own "comfortable" value. This value was calculated from the proximity value to the destination and the population density within the cell. It was calculated using equation (1). *Wden* and *Wdis* were weights. In our experiment *Wden* was 0.3 and *Wdis* was 0.7.

The population density told us how agents were spread out. This information helped maintaining the distance between agents.

$$Comfort = Wden*Density + Wdis*Distance \quad (1)$$

Density was a ratio between the number of agents in the cell and the maximum number of agents that could be put in that cell, as shown in equation (2).

$$Density = n / m \quad (2)$$

Where:
n is the number of agents in the cell
m is the maximum number of agents in the cell.

Once a comfortable value from each cell was obtained, each cell observed comfortable values from all its neighbors (8 neighbors).

When the system started, each cell was given its initial direction using array indices comparison between itself and the destination. This calculation is shown in Figure 2. The direction was mapped to a simplified angle shown in Figure 3. Once the system got going, each cell calculated its direction from average angles of its surrounding neighbors and its best comfortable direction (see equation (3)). Figure 4(a) shows a direction of the calculating cell (the middle cell) obtained from averaging out all its neighbors' directions. The top right cell was the most comfortable cell. Our equation effectively steer the calculating cell's direction towards the most comfortable cell (see Figure 4(b)). In our experiment, we used a fix ratio in the equation (W_{avg} was 0.3 and W_{com} was 0.7), but this could be changed according to experiment.

$$Angle = W_{avg} * averageAngle + W_{com} * comfortAngle \qquad (3)$$

If index i in current cell – index i in target cell = 0 then target cell at same row
Else If index i in current cell – index i in target cell < 0 then target cell at below row
Else If index i in current cell – index i in target cell > 0 then target cell at above row

If index j in current cell – index j in target cell = 0 then target cell at same column
Else If index j in current cell – index j in target cell < 0 then target cell at right row
Else If index j in current cell – index j in target cell > 0 then target cell at left row

Figure 2. Pseudo code for selecting the initial direction

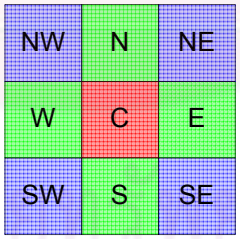


Figure 3. Initial Cell Direction

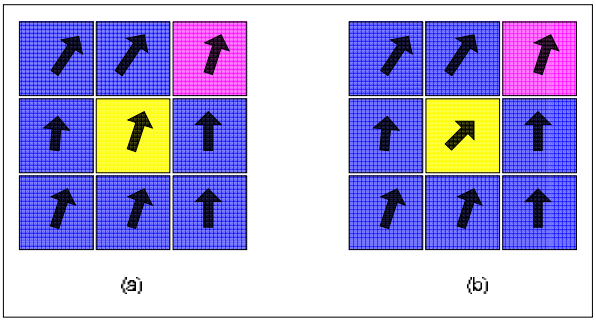


Figure 4. Cell Direction Calculation

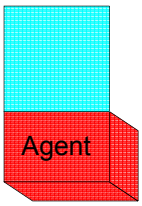


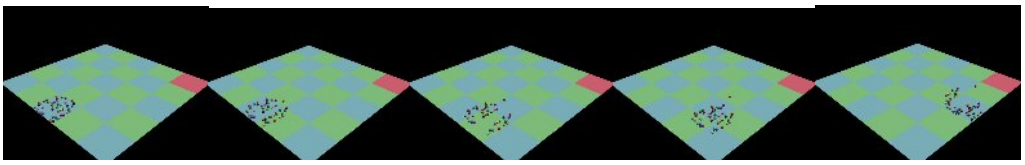
Figure 5. Agent's Sensor Region

After common decisions for all agents in a cell were made, agents moved according to the calculated direction and speed. Random variations in direction and speed were added to some agents in order to prevent agents from the same group from behaving too similarly. Agents avoided collision using a standard flocking procedure. Each agent had its limited sensor that could detect its surrounding. A vision of each agent is shown in Figure 5. If there was anything obstructing its path, an agent would gradually change its direction until it found an unobstructed path. Agents tried to keep a good distance from others when it reached its destination.

4. Experiment and Result

Our scenes were created using 3ds max. We ran our experiment using Intel(R) Pentium(R) M processor 1.60 GHz with 1278 MB of memory, running on window xp sp2 with DirectX 9.0c and GPU ATI Mobility Radeon 9700 64 MB. We ran seven scenarios. Each scenario involved agents moving from their starting cell(s) to a destination cell. Each scenario either had different starting cell(s), destination cells, or different number of agents. Most of our Cellular Flocking agents were able to move to their destination, as well as display the characteristics of ordinary flocking, such as avoiding collision, maintaining group proximity, and moving in angles influenced by other agents. Figure 6(a) - 6(e) show the movement of Cellular Flocking agents in a scene.

We ran our seven scenarios using both ordinary flocking and Cellular Flocking and compare their running time. The resource utilization of Cellular Flocking is shown in Table 1. The resource utilization of ordinary flocking is shown in Table 2. Starting points and destination points are referred to as numbers. The position in our map that each number identified is defined in Figure 7.



(a) (b) (c) (d) (e)

Figure 6. Cellular flocking behavior

1	2	3	4	5
6	7	8	9	10
11	12	13	14	15
16	17	18	19	20
21	22	23	24	25

Figure 7. Map position index

Table 1. Resource Utilization of Cellular Flocking

Cellular flocking						
Scenario	Number of agents	Starting Cell	Target Cell	cpu	Memory(MB)	Time(m:s)
1	50	11	25	100	750	2:11
2	50	6 11 16	25	100	751	2:19
3	50	6 11 16	5	100	753	2:00
4	50	1	25	100	661	2:11
5	50	6 11 16	15	100	613	2:08
6	100	11	25	100	650	2:54
7	200	11	25	100	750	4:16

Table 2. Resource Utilization of ordinary flocking

Ordinary flocking						
Scenario	Number of agents	Starting Cell	Target Cell	cpu	Memory(MB)	Time(m:s)
1	50	11	25	100	751	3:27
2	50	6 11 16	25	100	745	3:32
3	50	6 11 16	5	100	746	2:49
4	50	1	25	100	744	0:27

5	50	6 11 16	15	100	751	3:18
6	100	11	25	100	752	1:20
7	200	11	25	100	769	10:20

It can be seen from the tables that Cellular Flocking used less running time and memory for most cases. In larger crowds, the running time advantage could be extremely significant. In scene 4 and 7, ordinary flocking surprisingly appeared to perform better. We believed that this was due to many agents wandering off the map in those scenes, accidentally causing the reduction in computation for ordinary flocking.

5. Conclusion

This paper presented a new technique that reduced the flocking calculation to the level of cells in the map. Our result showed that the technique reduced the running time considerably. This new technique could be used to save production time in the animation and game industry.

References

[1] W. Tantisiriwat, A. Sumleeon, P. Kanongchaiyos: A Crowd Simulation Using Individual-Knowledge-Merge based Path Construction and Smoothed Particle Hydrodynamics. *In Proceedings of 15th International Conference in Central Europe on Computer Graphics, Visualization and Computer Vision'2007*, University of West Bohemia, Campus-Bory Plzen, Plezn, Czech Republic, January 29 - February 1, 2007.

[2] M. Aoyagi, A. Namatame: Massive Multi-Agent Simulation in 3D. *Advances in Soft Computing*. Springer Berlin / Heidelberg Volume 29/2005.

[3] R. O. Saber, R. M. Murray: Flocking with Obstacle Avoidance: Cooperation with Limited Communication in Mobile Networks. *In Proceeding of 42nd IEEE Conference on Decision and Control, 2003*, December 9-12, 2003.

[4] Leo Bi, R. Hall: Coherent Formaion for Agents Using Flocking with Cellular Automata. *Lecture Notes in Computer Science*. Springer Berlin / Heidelberg Volume 4251/2006.

[5] T. Hamagami, H. Hirata: Method of crowd simulation by using multiagent on cellular automata. *IEEE/WIC International Conference on Intelligent Agent Technology 2003 (IAT 2003),* October 13-16 2003.

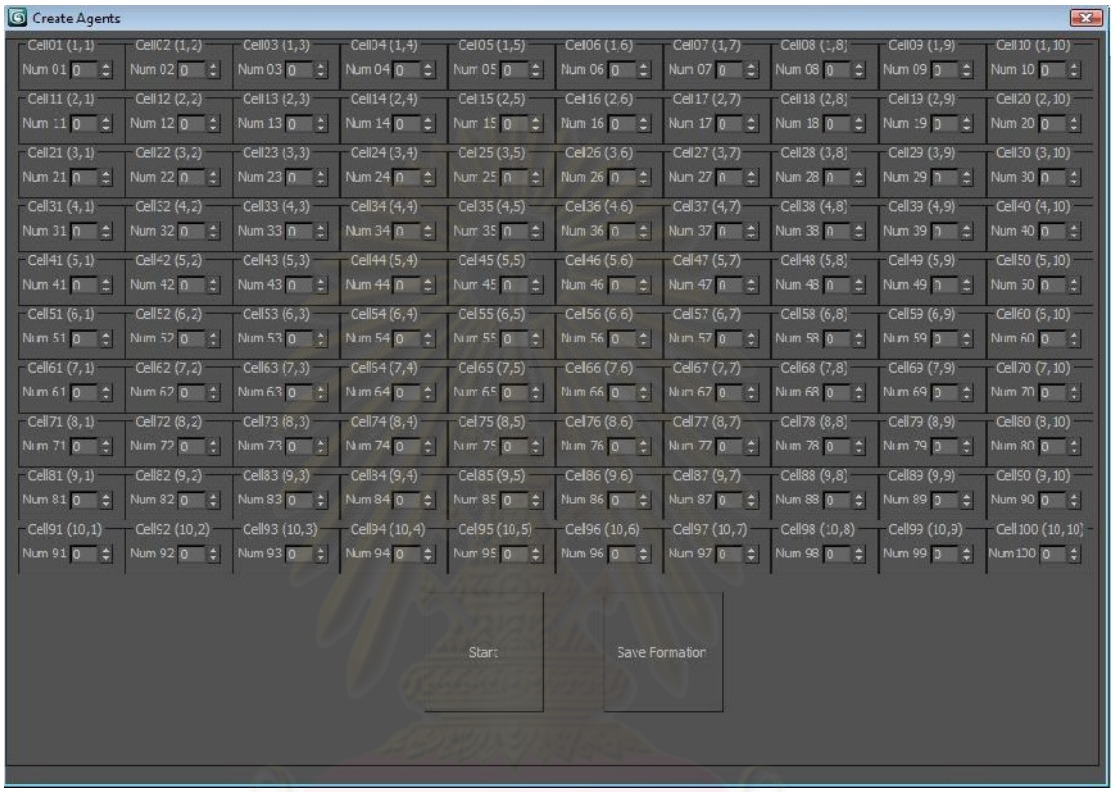
[6] S. Bandini, S. Manzoni, G. Vizzari: Situated Cellular Agents for Crowd simulation and Visualization. http://www.iemss.org/iemss2004/pdf/complexinteract/bandsitu.pdf (last accessed 15 March 2009).

[7] P. Sweetser, J. Wiles: Combining Influence Maps and cellular Automata for Reactive Game Agents. *6th International Conference on Intelligent Data Engineering and Automated Learning (IDEAL 2005)*, July 6-8, 2005.

[8] A. Treuille, S. Cooper, Z. Popovic: Continuum Crowds. *ACM SIGGRAPH 2006*, Volume 25, Issue 3, ACM New York, NY, USA July 2006.

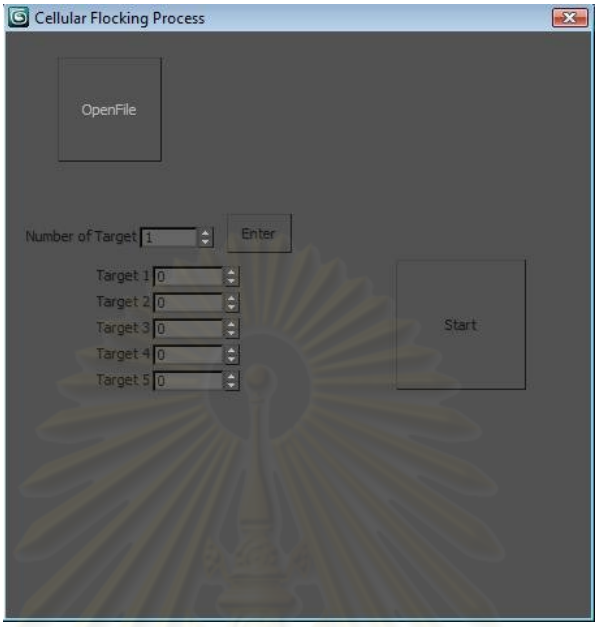
ภาคผนวก ข หน้าจอผู้ใช้งาน

หน้าจอโปรแกรมที่ใช้กำหนดตำแหน่งของเอเจนต์



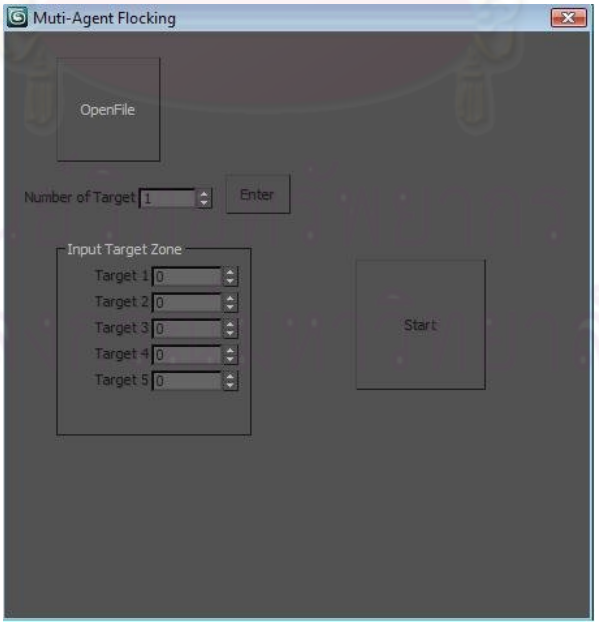
รูปที่ 60 แสดงหน้าจอผู้ใช้งานที่ใช้สร้างเอเจนต์ในตำแหน่งต่างๆ

หน้าจอโปรแกรมเซลลูลาร์ฟลอกกิง



รูปที่ 61 แสดงหน้าจอผู้ใช้งานของโปรแกรมเซลลูลาร์ฟลอกกิง

หน้าจอโปรแกรมมัลติเอเจนต์ฟลอกกิง



รูปที่ 62 แสดงหน้าจอผู้ใช้งานของโปรแกรมมัลติเอเจนต์ฟลอกกิง

ประวัติผู้เขียนวิทยานิพนธ์

ผู้เขียนสำเร็จการศึกษาระดับปริญญาวิทยาศาสตรบัณฑิต สาขาคณิตศาสตร์ จากภาควิชาคณิตศาสตร์ คณะวิทยาศาสตร์ มหาวิทยาลัยศิลปากรในปีการศึกษา 2547 หลังจากทำงานแล้วได้เข้าศึกษาต่อในหลักสูตรวิทยาศาสตรมหาบัณฑิต สาขาวิทยาศาสตร์คอมพิวเตอร์ คณะวิศวกรรมศาสตร์ จุฬาลงกรณ์มหาวิทยาลัยในปีการศึกษา 2549