บทที่ 2
แนวคิด ทฤษฎีและงานวิจัยที่เกี่ยวข้อง

การพัฒนาระบบการติดตามและแสดงตำแหน่งเรือประมงด้วยเทคโนโลยีความเป็นจริงเสริมนั้น จำเป็นต้องใช้ทฤษฎีหลายด้านเข้ามาช่วย เช่นหลักการทำงานและการเขียนโปรแกรมบนระบบปฏิบัติการ (operating system : OS) ที่ใช้ในอุปกรณ์เคลื่อนที่ (mobile device) วิธีการอ่านค่าพิกัดและทิศทางของอุปกรณ์ที่ผู้ใช้ถือหรือสวมใส่ การนำข้อมูลจากเซ็นเซอร์แหล่งต่าง ๆ มารวมกับเพื่อการตรึง (register) พิกัดของผู้ใช้กับพิกัดโลกจริง รวมทั้งเทคนิคในการเขียนโปรแกรมเพื่อนำภาพ กราฟิกส์มาซ้อนทับลงบนภาพจากโลกจริง ซึ่งในงานวิจัยนี้พัฒนาโปรแกรมบนระบบปฏิบัติการแอนดรอยด์ (Android) และใช้คลังโปรแกรมโอเพนจีแอล (Open Graphic Library : OpenGL) เป็นเครื่องมือหลักในการแสดงผลกราฟิกส์และความเป็นจริงเสริม โดยเนื้อหาส่วนที่เกี่ยวข้องกับการแปลงพิกัดและโอเพนจีแอลนั้นเรียบเรียงจากคัมภีร์ ธีระเวช (2557) และคัมภีร์และคณะ (2558) ซึ่งรายละเอียดของหลักการแต่ละส่วนมีดังนี้

2.1 ระบบปฏิบัติการแอนดรอยด์

แอนดรอยด์เป็นชื่อของระบบปฏิบัติการแบบโอเพนซอสซ์ (opensource) สำหรับอุปกรณ์เคลื่อนที่ พัฒนาโดยกูเกิล (Google) ใช้ชุดคำสั่งพื้นฐานและเคอร์เนล (kernel) ระบบปฏิบัติการลินุกซ์ (Linux) มีลิขสิทธิ์ในการใช้งานซอซส์โค๊ดแบบอาปาเชไลเซนส์ (Apache License) โดยหลักการพัฒนาโปรแกรมประยุกต์บนระบบปฏิบัติการแอนดรอยด์นั้นสามารถอ้างอิงได้จาก Google (2017) ซึ่งสามารถสรุปได้ดังนี้

2.1.1 การพัฒนาโปรแกรมบนระบบปฏิบัติการแอนดรอยด์

การพัฒนาโปรแกรมให้ทำงานบนแอนดรอยด์นั้นสามารถทำได้ด้วยภาษาคอมพิวเตอร์ต่าง ๆ ทั้งภาษาหลัก เช่น ซี (C) ซีพลัสพลัส (C++) จาวา (Java) และภาษาอื่น ๆ เช่น ค็อทลิน (Kotlin) โดยการเรียกใช้ฟังก์ชันต่าง ๆ จากชุดเครื่องมือพัฒนาโปรแกรมแอนดรอยด์หรือแอนดรอยด์เอสดีเค (Android SDK) ซึ่งโปรแกรมที่สามารถทำงานได้ (executeable) จะมีชื่อเรียกว่าเอพีเค (Android package : APK) หรือนิยมเรียกว่าแอป (app) โดยมีองค์ประกอบภายในแอปที่สำคัญจำนวนสี่ประเภท แต่ละประเภทมีหน้าที่และวัตถุประสงค์การใช้งานต่างกัน ดังนี้

1) แอคทีวิตี

แอคทีวิตี (activity) เป็นจุดเข้า (entry point) หรือส่วนของแอปที่ทำหน้าที่โต้ตอบกับผู้ใช้ โดยทั่วไปแอคทีวิตีหนึ่ง ๆ นั้นจะหมายถึงส่วนติดต่อผู้ใช้แบบกราฟิกส์หนึ่งหน้า โดยแอคทีวิตีจะทำหน้าที่ติดตามการโต้ตอบต่าง ๆ จากผู้ใช้ โดยเฉพาะข้อมูลจากโพรเซลที่รันก่อนหน้าหรือกำลังรัน

2) เซอร์วิส

เซอร์วิส (Services) เป็นบริการจุดเข้าแบบทั่วไป (general-purpose entry point) โดยใช้เพื่อการประมวลผลพื้นหลัง (background process) และการรันโปรแกรมพื้นหลังแบบระยะยาว และโดยมากจะไม่มีส่วนติดต่อผู้ใช้แบบกราฟิกส์ (graphical user interface : GUI)

3) บรอดแคสรีซีฟเวอร์

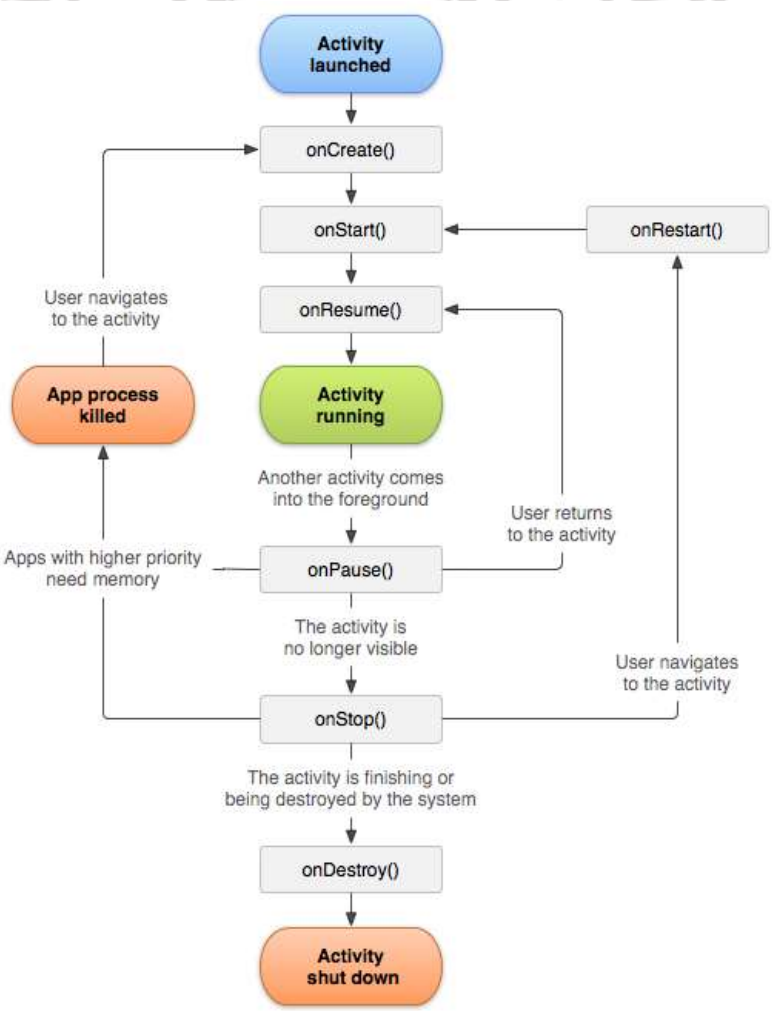
บรอดแคสรีซีฟเวอร์ (broadcast receivers) เป็นองค์ประกอบหนึ่งของแอปที่ช่วยให้ระบบสามารถส่งข้อมูลเหตุการณ์ระหว่างกันได้ ซึ่งสามารถใช้ส่งข้อมูลไปยังแอปที่ยังไม่ได้รันได้อีกด้วย ซึ่งสัญญาณบรอดแคสบางส่วนอาจจะมาจากระบบปฏิบัติการก็เป็นได้

4) คอนเท็นท์โพรไวเดอร์

คอนเท็นท์โพรไวเดอร์ (content providers) เป็นส่วนเสมือนตัวจัดการ (manager) ที่ช่วยให้ผู้ใช้จัดการข้อมูลของแอปเพื่อให้สามารถใช้ส่วนจัดเก็บข้อมูลของระบบปฏิบัติการแอนดรอยด์ได้

2.1.2 วัฏจักรชีวิตของแอคทีวิตี

แอคทีวิตีมีวัฏจักรชีวิต (lifecycle) ที่สำคัญดังแสดงในภาพที่ 2.1 โดยภาพแสดงการเปลี่ยนแปลงสถานะของแอคทีวิตีระหว่างที่มีการทำงานตลอดวัฏจักรชีวิต และมีการดักหรือฟัง (listen) เหตุการณ์และการเรียกกลับ (callback) ที่สำคัญดังนี้



ภาพที่ 2.1 วัฏจักรชีวิตของแอคทีวิตี
ที่มา: Google, 2017

1) เมท็อด `onCreate`

เมท็อด `onCreate` เป็นเมท็อดเรียกกลับที่จะทำงานเมื่อระบบมีการสร้างแอคทีวิตีขึ้นมา โดยในขั้นตอนการสร้างจะทำให้แอคทีวิตีเข้าสู่สถานะสร้างแล้ว (created) โดยโปรแกรมเมอร์จะใช้เมท็อดนี้เพื่อเตรียมหรือจัดการข้อมูลต่าง ๆ ที่จะถูกนำไปใช้ตลอดวัฏจักรชีวิตของแอคทีวิตี เช่น การเตรียมค่าโดยปริยายของตัวแปรและการเตรียมวิว (View) หรือคอนโทรลต่าง ๆ ให้พร้อมใช้งานตามที่ผู้ใช้ต้องการ

2) เมท็อด `onStart`

เมื่อแอคทีวิตีพร้อมทำงานจะเข้าสู่สถานะเริ่มต้นแล้ว (started) ซึ่งเมท็อด `onStart` จะถูกเรียกให้ทำงานและทำให้แอคทีวิตีปรากฏให้ผู้ใช้เห็น อย่างไรก็ตามสถานะเริ่มต้นแล้วนั้นไม่ใช่สถานะถาวรของแอคทีวิตี แต่จะเป็นเพียงสถานะชั่วคราวและเมื่อแอคทีวิตีทำงานต่าง ๆ ที่กำหนดในเมท็อด `onStart()` เสร็จสิ้นก็จะเปลี่ยนสถานะไปเป็นทำงานต่อ (resummed) ทันที

3) เมท็อด `onResume`

เมื่อแอคทีวิตีเปลี่ยนมาอยู่ในสถานะทำงานต่อ จะทำให้เมท็อด `onResume` ทำงานทันที โดยทั่วไปแล้วจะยังทำให้แอคทีวิตียังคงอยู่เป็นพื้นหน้าของโปรแกรมและสามารถโต้ตอบกับผู้ใช้ได้ ซึ่งแอคทีวิตีจะอยู่ในสถานะนี้ไปจนกว่าจะมีการเปลี่ยนแปลงที่ทำให้แอคทีวิตีเสียโฟกัส (lost focus) จากการใช้งาน เช่น มีการเรียกแอคทีวิตีอื่นทำงานหรือผู้ใช้เรียกใช้แอปอื่น เป็นต้น

4) เมท็อด `onPause`

ระบบจะเรียกเมท็อด `onPause` นี้ให้ทำงานเมื่อผู้ใช้มีการละทิ้งหรือทำให้ แอคทีวิตีหลุดจากโฟกัสการทำงานดังที่ได้กล่าวมาแล้ว ในสถานะพัก (paused) นี้จะทำให้แอคทีวิตีพ้นจากการทำงานพื้นหน้ามาเป็นการทำงานพื้นหลัง แต่ยังอาจจะปรากฏให้ผู้ใช้เห็นได้อยู่ ดังนั้นเมท็อดนี้จึงนำไปใช้เพื่อให้แอปตัดสินใจว่าจะดำเนินการใดต่อหรือหยุดการทำงานส่วนใดในระหว่างที่แอคทีวิตีหลุดจากโฟกัส

5) เมท็อด `onStop`

แอคทีวิตีจะเข้าสู่สถานะหยุดทำงาน (stopped) เมื่อแอคทีวิตีถูกซ้อนทับด้วยแอคทีวิตีอื่นอย่างสิ้นเชิง หรือไม่ปรากฏการทำงานให้ผู้ใช้เห็นในพื้นหน้า และเมท็อด `onStop` จะถูกเรียกให้ทำงานเพื่อให้ระบบสามารถยุติการทำงานของระบบบางส่วนที่ไม่จำเป็นเพื่อประหยัดทรัพยากรของอุปกรณ์ได้

6) เมท็อด `onDestroy`

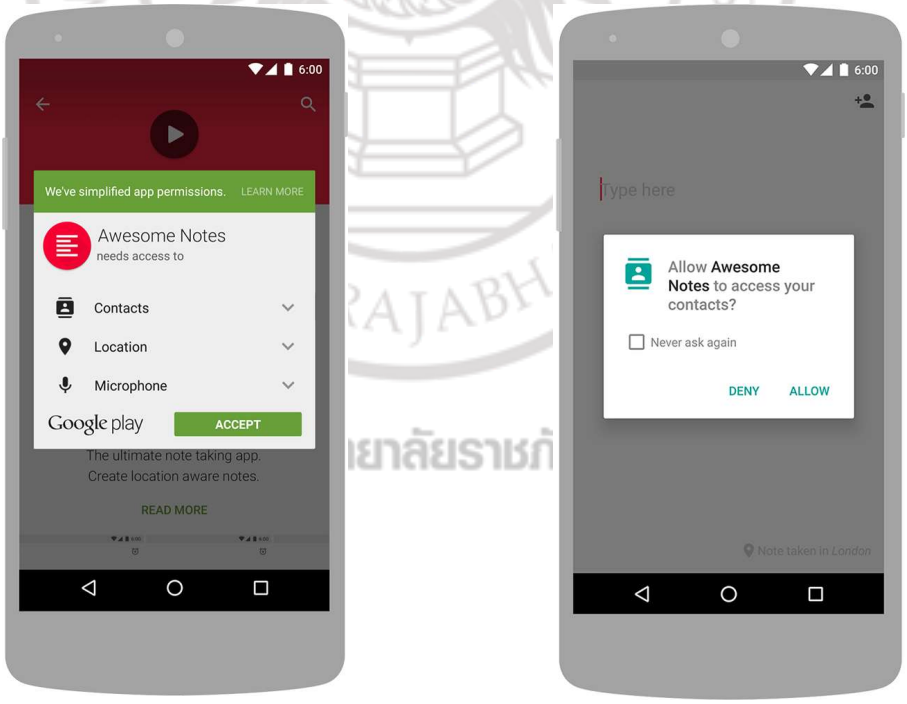
เมท็อดนี้จะทำงานเมื่อแอคทีวิตีถูกทำลาย (destroyed) ซึ่งอาจจะเกิดขึ้นได้เมื่อแอคทีวิตีสิ้นสุดการทำงานด้วยตัวเอง ผู้ใช้ยุติการทำงาน และเมื่อระบบสั่งยุติการทำงานเนื่องจากมีการเปลี่ยนแปลงบางอย่างเกิดขึ้นกับอุปกรณ์ เช่น เมื่อผู้ใช้เปลี่ยนทิศทางการเอียงตัว (orientation) ของสมาร์ทโฟนจากแนวตั้ง (portrait) เป็นแนวนอน (landscape) เป็นต้น

2.1.3 การอนุญาต

เนื่องจากระบบปฏิบัติการแอนดรอยด์ทำงานบนสมาร์ทโฟนที่มีอุปกรณ์ติดตั้งไว้หลายประเภท อุปกรณ์บางชนิดอาจจะใช้งานได้โดยไม่ได้ก่อให้เกิดอันตรายหรือละเมิดความเป็นส่วนตัว (privacy) ของผู้ใช้ แต่อุปกรณ์บางอย่างอาจจะมีโอกาสเสี่ยงที่จะละเมิดความเป็นส่วนตัวได้

ยกตัวอย่างเช่น ไมโครโฟนและกล้อง ซึ่งหากแอปสามารถเข้าถึงและใช้งานได้อย่างอัตโนมัติก็จะทำให้ข้อมูลต่าง ๆ ของผู้ใช้งานรั่วไหลได้ ดังนั้นการเข้าถึงอุปกรณ์หรือบริการบางอย่างบนระบบปฏิบัติการแอนดรอยด์จึงต้องรอให้มีการอนุญาต (permission) จากผู้ใช้ก่อน แอปจึงจะสามารถเข้าถึงและใช้งานอุปกรณ์หรือบริการที่ต้องการได้ ซึ่งการขออนุญาตนี้สามารถทำได้แบบอัตโนมัติและแบบที่มีการโต้ตอบจากผู้ใช้ การร้องขอเพื่อเข้าถึงอุปกรณ์และบริการเหล่านี้อาจจะมีเกิดขึ้นได้ตั้งแต่ขั้นตอนการติดตั้งโปรแกรม, การเริ่มต้นใช้งานโปรแกรม หรือระหว่างการใช้งานก็เป็นได้ ในเบื้องต้นหากโปรแกรมต้องการเข้าถึงอุปกรณ์และบริการที่จำเป็นต่อการใช้งานนั้น โปรแกรมจะต้องประกาศการขออนุญาตด้วยการใช้แท็ก `<uses-permission>` ในแฟ้มเมนิเฟสต์ (manifest) หากระบบพบว่ารายการขออนุญาตนั้นไม่ได้เป็นความเสี่ยงต่อความเป็นส่วนตัวของผู้ใช้มากเกินไปก็จะดำเนินการอนุญาตให้โดยอัตโนมัติ

อย่างไรก็ตามหากระบบพบว่าการใช้งานอุปกรณ์หรือบริการที่ร้องขอนั้นมีความเสี่ยงสูงแล้ว ระบบจะให้ผู้ใช้ยืนยันการอนุญาตให้มีการใช้งานผ่านกล่องโต้ตอบด้วยตัวเอง โดยในกรณีของ Android API 22 หรือต่ำกว่าจะมีแจ้งรายการการเข้าถึงอุปกรณ์หรือบริการต่างๆ ในขณะก่อนการติดตั้งโปรแกรม ดังแสดงตัวอย่างในภาพที่ 2.2 (ก) แต่สำหรับ Android API 23 หรือสูงกว่านั้นจะไม่มีการแจ้งเตือนการขออนุญาตในขั้นตอนการติดตั้งโปรแกรม แต่จะให้โปรแกรมขออนุญาตในขณะทำงานโดยใช้กล่องโต้ตอบของระบบเพื่อรอการอนุมัติการเข้าถึงจากผู้ใช้ ดังตัวอย่างในภาพที่ 2.2 (ข) ซึ่งแสดงตัวเลือกไม่ต้องถามอีก (never ask again) ซึ่งผู้ใช้สามารถเลือกให้ระบบปฏิบัติการจำคำตอบไว้เพื่อจะได้ไม่มีการถามซ้ำอีกในภายหลัง



(ก) (ข)

ภาพที่ 2.2 ตัวอย่างการขออนุญาตเข้าถึงบริการและอุปกรณ์ต่าง ๆ ของแอนดรอยด์
ที่มา: Google, 2017

ระบบปฏิบัติการแอนดรอยด์แบ่งระดับของการอนุญาตเพื่อปกต้องความเป็นส่วนตัวของผู้ใช้เป็นสี่ระดับ ดังนี้

1) การอนุญาตปกติ

การอนุญาตปกติ (normal permissions) เป็นการป้องกันระดับพื้นฐานที่จะครอบคลุมการเข้าถึงข้อมูลหรือทรัพยากรต่าง ๆ ของเครื่องในที่อยู่ภายนอกแซนด์บ๊อกซ์ (sandbox) ซึ่งมีความเสี่ยงที่จะละเมิดความเป็นส่วนตัวได้ต่ำ ยกตัวอย่างเช่น การกำหนดรูปแบบภาษา และการกำหนดวันที่และเวลา เป็นต้น

2) การอนุญาตแบบมีใบอนุญาต

การอนุญาตแบบมีใบอนุญาต (signature permissions) ระบบจะอนุญาตให้โปรแกรมเข้าถึงอุปกรณ์หรือบริการได้ก็ต่อเมื่อแอปที่ร้องขอนั้นมีใบอนุญาต (certificate) ชุดเดียวกันกับโปรแกรมที่เป็นผู้กำหนดสิทธิการเข้าถึงเท่านั้น ตัวอย่างของการอนุญาตในกลุ่มนี้คือบริการจากแอปภายนอก (third-party app)

3) การอนุญาตอันตราย

การอนุญาตอันตราย (dangerous permissions) เป็นการขออนุญาตที่มีความเสี่ยงที่จะมีอันตราย ซึ่งจะครอบคลุมการเข้าถึงข้อมูลหรือทรัพยากรของระบบที่เกี่ยวข้องกับข้อมูลความเป็นส่วนตัวของผู้ใช้ รวมทั้งการเข้าถึงข้อมูลหรือทรัพยากรที่อาจจะทำให้เกิดความเสียหายกับระบบปฏิบัติการหรือโปรแกรมอื่น ตัวอย่างเช่น การเข้าถึงรายชื่อใน สมุดโทรศัพท์ (contacts) เป็นต้น หากโปรแกรมใดต้องการเข้าถึงการป้องกันในระดับนี้จะต้องทำการขออนุญาตในขณะทำงานเท่านั้น

4) การอนุญาตพิเศษ

การอนุญาตพิเศษ (special permissions) จะเกิดขึ้นหากการดำเนินการบางอย่างอาจจะไม่ได้เข้าข่ายการละเมิดแบบใดแบบหนึ่งตามที่ได้กล่าวมา ยกตัวอย่างเช่น ขออนุญาตเพื่อการแก้ไขการตั้งค่า (`WRITE_SETTINGS`) เป็นต้น ในกรณีนี้โปรแกรมจะต้องกำหนดค่าในเมนิเฟสต์และจะต้องสร้างอินเทนท์ (intent) เพื่อร้องขอการอนุญาตจากผู้ใช้

2.2 ระบบกำหนดตำแหน่งบนโลก

ระบบกำหนดตำแหน่งบนโลกหรือจีพีเอส (global positioning system : GPS) เป็นระบบหรือบริการของประเทศสหรัฐอเมริกาที่ให้บริการด้านการกำหนดตำแหน่ง (positioning), การนำหนหรือการนำทาง (navigation) และการกำหนดเวลา (timing) (USAF, 2017) มีรายละเอียดที่ควรทราบดังนี้

2.2.1 องค์ประกอบ

ระบบจีพีเอสประกอบด้วยองค์ประกอบหลักที่เรียกว่าส่วน (segment) จำนวนทั้งหมดสามส่วน ประกอบด้วยส่วนอวกาศ (space segment) ส่วนควบคุม (control segment) และ ส่วนผู้ใช้ (user segment) โดยกองทัพอากาศสหรัฐอเมริกา (U.S. Air Force : USAF) มีหน้าที่พัฒนา บำรุงรักษาและดำเนินการในส่วนของส่วนอวกาศและส่วนควบคุม รายละเอียดของแต่ละส่วนมีดังนี้

1) ส่วนอวกาศ

องค์ประกอบส่วนอวกาศของระบบจีพีเอสประกอบด้วยดาวเทียมในวงโคจรแบบวงโคจรระยะปานกลางหรือเอ็มอีโอ (medium Earth orbit : MEO) จำนวนอย่างน้อย 24 ดวงจากจำนวนอย่าง 31 ดวงที่สามารถทำหน้าที่ได้ตลอดเวลา กลุ่มของดาวเทียมนี้จะแบ่งกันตามระนาบโคจร (orbital plane) ได้เป็นหกระนาบในแต่ละระนาบจะมีดาวเทียมทำงานประมาณสี่ดวง การจัดเรียงของดาวเทียมลักษณะจะทำให้แน่ใจได้ว่าผู้ใช้งานจะสามารถได้รับข้อมูลจากดาวเทียมอย่างน้อยสี่ดวง ณ ตำแหน่งใด ๆ บนโลกได้ตลอดเวลา

ดาวเทียมจีพีเอสแต่ละดวงทำหน้าที่ส่งหรือกระจายสัญญาณผ่านคลื่นวิทยุจำนวนทั้งหมดสองความถี่ในช่วงความถี่แอลแบนด์ (L-band) คือความถี่ 1,575.42 เมกะเฮิรตซ์ เรียกว่าแอลวัน (L1) และ 1,227.60 เมกะเฮิรตซ์ เรียกว่าแอลทู (L2) ซึ่งมีความยาวคลื่นประมาณ 19 เซนติเมตร และ 24.4 เซนติเมตร ตามลำดับ โดยดาวเทียมจีพีเอสแต่ละดวงจะมีความถี่คลื่นพาห์ (carrier frequency) เหมือนกันแต่ใช้เทคนิคการกล้ำสัญญาณ (modulation) ต่างกันเพื่อให้เกิดการแทรกสอดกันน้อยที่สุด สัญญาณจากดาวเทียมจีพีเอสจะแบ่งออกเป็นสองกลุ่มคือกลุ่มซีเอโค๊ด (coarse acquisition code : C/A-code) ซึ่งมีความถูกต้องของการบอกตำแหน่งต่ำกว่าสัญญาแบบพีโค๊ด (precision code : P-code) โดยสัญญาณซีเอโค๊ดจะถูกกล้ำเข้ากับความถี่คลื่นพาห์แอลวันเท่านั้น ในขณะที่สัญญาณพีโค๊ดจะถูกกล้ำเข้ากับคลื่นแอลวันและแอลทู ซึ่งเรียกว่าการกล้ำสองเฟสทวิภาค (binary biphase modulation) (El-Rabbary, 2002, pp.14) ตารางที่ 2.1 แสดงข้อมูลของดาวเทียมจีพีเอสรุ่นต่าง ๆ

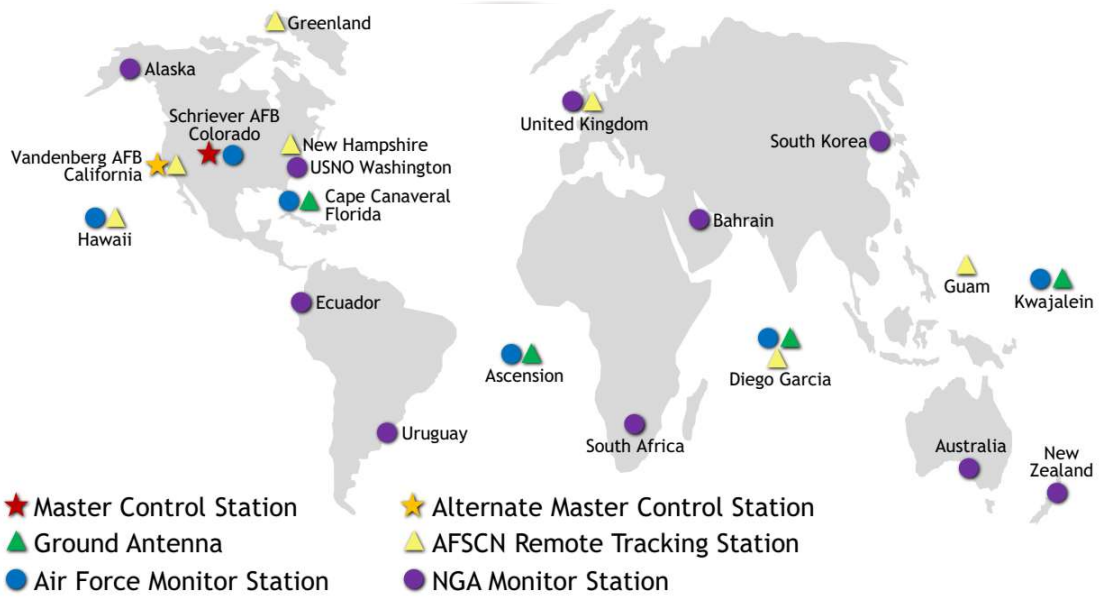
ตารางที่ 2.1 คุณสมบัติของดาวเทียมจีพีเอสรุ่นต่าง ๆ
ที่มา: USAF, 2018

รุ่น	จำนวน (ดวง)	ปีที่ขึ้นสู่วงโคจร (ค.ศ.)	อายุการใช้งาน (ปี)
BLOCK IIA	1	1990 - 1997	7.5
BLOCK IIR	11	1997 - 2004	7.5
BLOCK IIR-M	7	2005 - 2009	7.5
BLOCK IIF	12	2010 - 2016	12
BLOCK III/IIIF	ไม่มี	2018 - ปัจจุบัน	15

2) ส่วนควบคุม

ส่วนควบคุมประกอบด้วยเครือข่ายของสถานีภาคพื้นดินที่กระจายอยู่ทั่วโลก สถานีเหล่านี้แบ่งออกเป็นสถานีควบคุมหลัก (master control station : MCS) จำนวนหนึ่งสถานี อยู่ที่มลรัฐโคโรลาโดสปริงส์ ประเทศสหรัฐอเมริกา มีหน้าที่หลักเพื่อควบคุมการทำงานของดาวเทียม เช่น การรักษาระดับวงโคจร การตรวจสอบสถานะของดาวเทียม และวิเคราะห์ข้อมูลต่าง ๆ เป็นต้น โดยมีสถานีควบคุมหลักสำรอง (alternate master control station) จำนวนหนึ่งแห่งอยู่ที่รัฐ

แคลิฟอร์เนีย โดยสถานีควบคุมหลักจะสื่อสารกับดาวเทียมด้วยความถี่ในช่วงเอสแบนด์ (S-band) สำหรับสถานีอื่น ๆ เรียกว่าสถานีติดตาม (monitor station : MS) จำนวนห้าสถานีกระจายอยู่ทั่วโลก สถานีติดตามเหล่านี้ทำงานแบบอัตโนมัติผ่านการสั่งงานระยะไกลจากสถานีหลัก (El-Rabbary, 2002, pp.7) ภาพที่ 2.3 แสดงตำแหน่งของสถานีควบคุมและที่ตั้งเสาอากาศของระบบจีพีเอส



ภาพที่ 2.3 ข้อมูลสถานีควบคุมและที่ตั้งเสาอากาศของระบบจีพีเอส
ที่มา: USAF, 2017

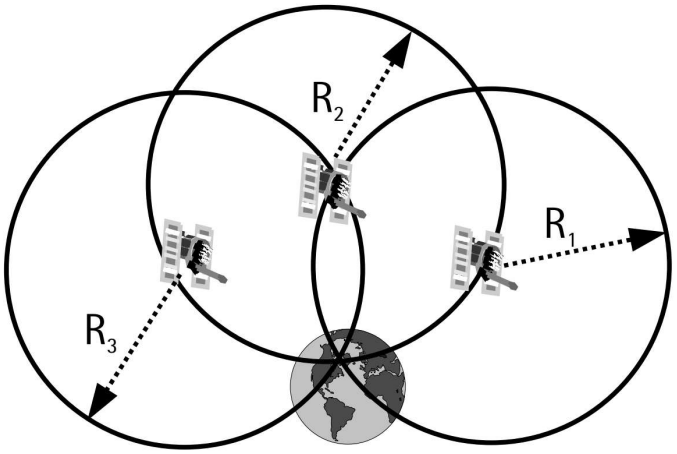
3) ส่วนผู้ใช้

ส่วนผู้ใช้แยกเป็นผู้ใช้พลเรือน (civilian users) และผู้ใช้ทหาร (military) โดยมีเครื่องรับสัญญาณจีพีเอส (GPS receiver) ทำหน้าที่รับและแสดงข้อมูลให้ผู้ใช้ทราบ โดยบริการส่วนนี้เป็นบริการฟรีและเปิด (USAF, 2017) เพื่อให้เกิดการพัฒนาและประยุกต์ได้อย่างกว้างขวาง

2.2.2 หลักการทำงาน

หลักการพื้นฐานของการระบุตำแหน่งด้วยจีพีเอสคือการวัดระยะทาง (distance) ระหว่างเครื่องรับสัญญาณจีพีเอสกับดาวเทียมจีพีเอส โดยในการคำนวณตำแหน่งนั้นจะต้องใช้ข้อมูลจากดาวเทียมอย่างน้อยสามดวงจึงจะสามารถประมาณตำแหน่งของเครื่องรับสัญญาณได้ โดยเทคนิคการคำนวณระยะทางเพื่อคำนวณตำแหน่งในระบบจีพีเอสนั้นมีสองแบบ คือการวัดระยะทางแบบระยะเทียมหรือซูโดเรนจ์ (pseudorange) และแบบเฟสคลื่นพาห์ (carrier-phase) โดยการวัดระยะแบบซูโดเรนจ์เป็นเทคนิคการหาตำแหน่งด้วยการประมาณระยะทางระหว่างดาวเทียมจีพีเอสและเครื่องรับสัญญาณจากดาวเทียมจีพีเอส ซึ่งระยะทางนี้เป็นระยะทางเทียมเนื่องจากไม่สามารถวัดระยะทางที่แท้จริงได้ ดังนั้นการคำนวณจึงจะใช้การนำค่าพารามิเตอร์ต่าง ๆ ของดาวเทียมร่วมกับการคำนวณระยะเวลาที่ใช้ในการส่งสัญญาณจากดาวเทียมมาถึงเครื่องรับสัญญาณ อย่างไรก็ตามปัญหาประการหนึ่งของการวัดระยะทางแบบนี้ความคลาดเคลื่อนในการวัดเวลา ซึ่งความคลาดเคลื่อนนี้มีผล

หลักมาจากความแม่นยำของตัวจับสัญญาณนาฬิกาที่ใช้ในเครื่องรับสัญญาณ ภาพที่ 2.4 แสดงแนวคิดการคำนวณตำแหน่งด้วยการวัดระยะทาง

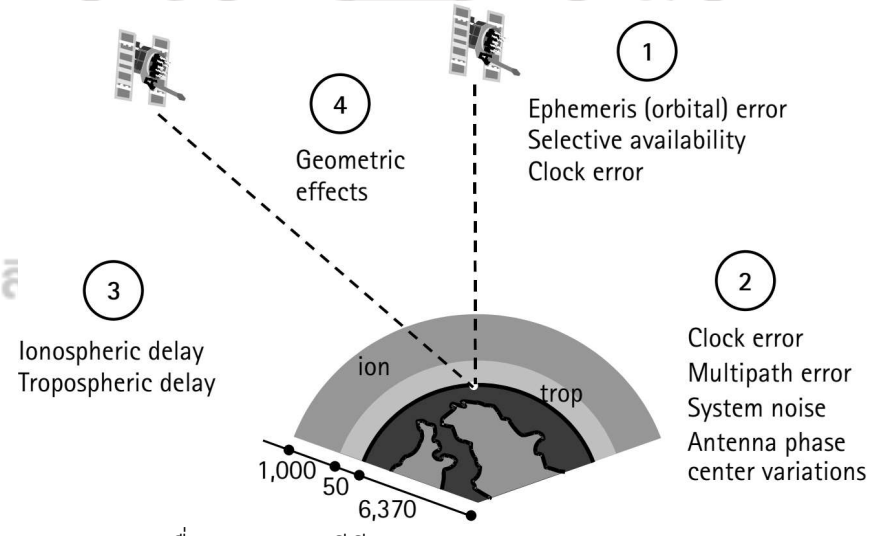


ภาพที่ 2.4 การกำหนดตำแหน่งด้วยดาวเทียม
ที่มา: El-Rabbany, 2002, pp.28

สำหรับการคำนวณระยะแบบเฟสคลื่นพาห์จะคำนวณการเปลี่ยนแปลงหรือการเปรียบเทียบเฟสของสัญญาณที่ส่งมาจากดาวเทียมจีพีเอสกับสัญญาณของเครื่องรับ โดยจะนำจำนวนลูกคลื่นที่ส่งจากดาวเทียมมาถึงเครื่องรับและความยาวคลื่นพาหะที่ใช้ในการสื่อสารมาคำนวณก็จะได้ระยะทางระหว่างดาวเทียมกับเครื่องรับ การวัดระยะแบบมีความแม่นยำมากกว่าการวัดแบบซูโดเรนจ์

2.2.3 ข้อจำกัด

พิกัดที่ได้จากเครื่องรับสัญญาณจีพีเอสนั้นจะมีคลาดเคลื่อนเกิดขึ้นได้ตลอดเวลา ซึ่งอาจจะเนื่องมาจากหลายปัจจัย ภาพที่ 2.5 แสดงตัวอย่างแหล่งกำหนดความคลาดเคลื่อนของจีพีเอส



ภาพที่ 2.5 ความคลาดเคลื่อนของระบบจีพีเอส
ที่มา: El-Rabbany, 2002, pp.28

ยกตัวอย่างเช่น ความคลาดเคลื่อนเนื่องจากการเคลื่อนที่ของสัญญาณแม่เหล็กไฟฟ้าที่เคลื่อนที่ผ่านชั้นบรรยากาศของโลก ความคลาดเคลื่อนเนื่องจากวงโคจรของดาวเทียม ซึ่งตำแหน่งของดาวเทียมในวงโคจรอาจจะมีการเปลี่ยนแปลงหรือแตกต่างจากตำแหน่งที่คำนวณจากค่าพารามิเตอร์ที่กำหนดไว้ล่วงหน้าได้ และความคลาดเคลื่อนเนื่องจากอุปกรณ์จับเวลาที่ติดตั้งไว้บนดาวเทียม เป็นต้น

ความคลาดเคลื่อนที่ผู้ใช้พลเรือนมักพบได้เสมอคือความคลาดเคลื่อนหลายเส้นทาง (multipath errors) หรือความคลาดเคลื่อนเนื่องจากสัญญาณสะท้อน (reflected signal) ซึ่งเกิดจากสัญญาณจีพีเอสจากชั้นบรรยากาศเดินทางมากระทบกับวัตถุอื่นก่อนที่จะสะท้อนเข้าหาเครื่องรับสัญญาณจีพีเอส การสะท้อนนี้อาจจะเกิดขึ้นได้มากกว่าหนึ่งครั้ง ตัวอย่างที่มักพบได้เสมอคือสัญญาณจากอวกาศเดินทางผ่านชั้นบรรยากาศเข้ามากระทบพื้นดินแล้วสะท้อนไปยังอาคารที่อยู่รอบ ๆ เครื่องรับสัญญาณจีพีเอสก่อนที่จะสะท้อนเข้าหาเครื่องสัญญาณจีพีเอส โดยในกรณีของเขตเมืองที่มีอาคารสูงนั้นมักจะพบความคลาดเคลื่อนเช่นนี้ได้ง่ายเนื่องจากความสูงของอาคารทำให้เกิดการสะท้อนได้หลายครั้ง

2.2.4 ระบบดาวเทียมนำทางบนโลก

ระบบดาวเทียมนำทางบนโลกหรือจีเอ็นเอสเอส (global navigation satellite systems : GNSS) เป็นชื่อโดยทั่วไปของระบบนำทางที่ใช้ดาวเทียมเพื่อการระบุตำแหน่งและเวลาเช่นเดียวกับระบบจีพีเอสของประเทศสหรัฐอเมริกา แต่มีการบริหารและจัดการโดยประเทศอื่นบางระบบบริหารงานโดยองค์การระหว่างประเทศ ตารางที่ 2.2 แสดงระบบดาวเทียมที่ทำหน้าที่ในกลุ่มนี้ โดยในจำนวนนี้มีระบบเป่ยโต่ว (BeiDou), กาลิเลโอ (Galileo) และโกลนาส (GLONASS) ที่ให้บริการครอบคลุมพื้นที่ทั้งโลก ในขณะที่ไออาร์เอ็นเอสเอส (IRNSS) นั้นบริการในระดับภูมิภาคบริเวณประเทศอินเดียและพื้นที่อื่นภายในรัศมีประมาณ 1,500 กิโลเมตรจากประเทศอินเดีย สำหรับคิวซีเอสเอส (QZSS) นั้นเป็นบริการระดับภูมิภาคของประเทศญี่ปุ่นที่ให้บริการพื้นที่แถบเอเชียตะวันออกและแถบโอเชียเนีย (Ocienia)

ตารางที่ 2.2 ระบบดาวเทียมนำทางบนโลก
ที่มา: USAF, 2018

ชื่อ	เจ้าของ	จำนวนดาวเทียม
เป่ยโต่ว	จีน	35 ดวง ภายในปี ค.ศ. 2020
กาลิเลโอ	สหภาพยุโรป	อย่างน้อย 24 ดวง ภายในปี ค.ศ. 2020
โกลนาส	รัสเซีย	อย่างน้อย 24 ดวง
ไออาร์เอ็นเอสเอส	อินเดีย	7 ดวง ภายในปี 2018
คิวซีเอสเอส	ญี่ปุ่น	7 ดวง ภายในปี 2023

2.3 โอเพนจีแอลและคลังโปรแกรมที่เกี่ยวข้อง

คัมภีร์ ธีระเวช (2557) และคัมภีร์และคณะ (2558) ได้สรุปข้อมูลเกี่ยวกับโอเพนจีแอลไว้ว่า โอเพนจีแอลเป็นคลังโปรแกรม (library) แบบส่วนต่อประสานโปรแกรมประยุกต์หรือเอพีไอ (application programming interface : API) ที่ออกแบบมาเพื่อการสร้างภาพกราฟิกส์ รองรับการทำงานได้หลายภาษาและหลายระบบปฏิบัติการ โดยทางบริษัทเอสจีไอ (Silicon Graphics Inc. : SGI) ได้เริ่มต้นพัฒนาและเผยแพร่โอเพนจีแอลในช่วงต้นทศวรรษที่เก้าสิบ ซึ่งในช่วงนั้นใช้งานในกลุ่มการออกแบบใช้คอมพิวเตอร์ช่วย (computer aided design : CAD) และงานวิทยาศาสตร์/วิศวกรรมเป็นหลัก เวอร์ชันปัจจุบันรองรับการทำงานร่วมกับหน่วยประมวลผลกราฟิกส์หรือจีพียู (graphic processing unit : GPU) ซึ่งสามารถเร่งความเร็วในการประมวลผลข้อมูลได้มาก ส่งผลให้ประสิทธิภาพของโปรแกรมเพิ่มขึ้นตามไปด้วย ภาพที่ 2.6 (ก) แสดงโลโก้ของโอเพนจีแอล

(ก) โอเพนจีแอล (ข) โอเพนจีแอลอีเอส (ค) เว็บจีเอล

ภาพที่ 2.6 โลโก้ของโอเพนจีแอลและคลังโปรแกรมที่เกี่ยวข้อง
ที่มา: Khronos, 2018

คุณลักษณะอย่างหนึ่งของโอเพนจีแอลก็คือ โอเพนจีแอลนั้นเป็นเอพีไอแบบไม่ขึ้นกับฮาร์ดแวร์ (hardware independence) อย่างแท้จริง (Khronos, 2018) นอกจากนั้นการ์ดแสดงผลกราฟิกส์ส่วนใหญ่ในปัจจุบันก็มักจะมีการอิมพลิเม้นต์โอเพนจีแอลบนฮาร์ดแวร์เรียบร้อยแล้ว จึงทำให้โปรแกรมเมอร์สามารถพัฒนาโปรแกรมกราฟิกส์ที่สามารถใช้งานกราฟิกส์การ์ดได้หลายรุ่น จุดเด่นอีกประการหนึ่งของโอเพนจีแอลคือสิ่งที่เรียกว่าส่วนต่อขยาย (extension) ซึ่งทำให้ผู้ผลิตการ์ดแสดงผลกราฟิกส์สามารถเพิ่มฟังก์ชันต่าง ๆ ตามความต้องการของตัวเองเข้ากับโอเพนจีแอลได้อย่างอิสระ แนวคิดของส่วนต่อขยายนี้ทำให้โปรแกรมเมอร์สามารถใช้งานโอเพนจีแอลได้อย่างมีประสิทธิภาพและสามารถสร้างสรรค์ภาพกราฟิกส์ได้มากกว่าการใช้ฟังก์ชันตามปกติ อย่างไรก็ตามเพื่อให้การใช้งานส่วนต่อขยายนี้ไม่กระทบต่อการใช้งานฟังก์ชันหลักของโอเพนจีแอล ส่วนต่อขยายต่าง ๆ นี้จะไม่ถูกรวมเข้ากับแกนภาษา (core language) ของโอเพนจีแอล แต่จะถูกไปรวบรวมผ่านทาง OpenGL Extension Registry[1] ซึ่งจะมีผู้ผลิตฮาร์ดแวร์กราฟิกส์หลายรายมาลงทะเบียนผลิตภัณฑ์ของตนเองไว้

2.3.1 การทำงานแบบเครื่องจักรสถานะ

ลักษณะสำคัญอีกประการหนึ่งของโอเพนจีแอลคือลักษณะการทำงานแบบเครื่องจักรสถานะ (state machine) นั่นคือเมื่อโปรแกรมเมอร์กำหนดให้โปรแกรมทำงานในสถานะใดหรืออยู่ในโหมดใดแล้วโอเพนจีแอลก็จะคงอยู่ในสถานะนั้นต่อไปจนกว่าผู้ใช้จะมีการกำหนดให้เป็นอย่างอื่น ตัวอย่างเช่น การกำหนดสีที่จะใช้วาดวัตถุ เป็นต้น ในบางกรณีเราสามารถกำหนดสถานะของโอเพนจีแอลได้ด้วยฟังก์ชัน `glEnable()` และปิดสถานะได้ด้วยการเรียกใช้งานฟังก์ชัน `glDisable()`

1 http://www.opengl.org/registry/

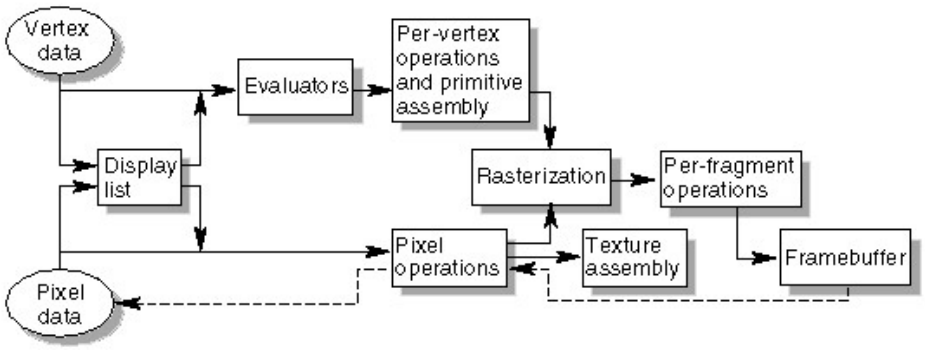
โดยส่งอาร์กิวเม้นต์ไปให้ฟังก์ชัน ค่าสถานะเหล่านี้จะบันทึกอยู่ในตัวแปรสถานะของโอเพนจีแอล ตารางที่ 2.3 แสดงตัวกรอง (mask) ของกลุ่มตัวแปรสถานะที่สำคัญบางค่าของโอเพนจีแอล

ตารางที่ 2.3 ตัวกรองของกลุ่มตัวแปรสถานะที่สำคัญบางค่าของโอเพนจีแอล
ที่มา: Shreiner, 2013

ชื่อตัวกรอง	กลุ่มของสถานะที่บันทึก
GL_COLOR_BUFFER_BIT	บัฟเฟอร์สี
GL_DEPTH_BUFFER_BIT	บัฟเฟอร์ความลึก
GL_ENABLE_BIT	การเปิดใช้งาน
GL_LINE_BIT	เส้น
GL_LIGHTING_BIT	แสง
GL_POINT_BIT	จุด
GL_POLYGON_BIT	โพลีกอน
GL_TRANSFORM_BIT	การแปลง

2.3.2 เรนเดอร์ริงไปป์ไลน์

เรนเดอร์ริงไปป์ไลน์ (rendering pipeline) เป็นชื่อที่ใช้เรียกขั้นตอนในแปลงชุดคำสั่งโอเพนจีแอลให้เป็นปรากฏบนจอภาพ โดยเรนเดอร์ริ่งไปป์ไลน์แบบคลาสสิก (classic rendering pipeline) มีรูปแบบตามภาพที่ 2.7 โดยข้อมูลอินพุตพื้นฐานของโอเพนจีแอลคือข้อมูลของวัตถุที่จะวาด ส่วนใหญ่คือจุดยอดหรือเวอร์เท็กซ์ (vertex) ซึ่งเป็นพิกัดของจุดยอดของรูปทรงต่าง ๆ ในกรณีของการสร้างภาพพื้นผิว (surface) จากรายการของจุดควบคุม (control points) โอเพนจีแอลจะต้องส่งพิกัดของจุดควบคุมเหล่านี้ไปให้ตัวประเมินผล (evaluator) ซึ่งเป็นฟังก์ชันการกำหนดแบบพหุนาม (polynomial mapping) สำหรับการคำนวณเวกเตอร์ปกติ (normal vector), พิกัดลายผิว (texture coordinates) รวมทั้งพิกัดสามมิติของจุดยอดของพื้นผิวที่ต้องการ



ภาพที่ 2.7 เรนเดอร์ริงไปป์ไลน์แบบคลาสสิก
ที่มา: Neider et al. (1999)

โดยข้อมูลกลุ่มนี้จะวิ่งผ่านตัวดำเนินการต่าง ๆ ตามเส้นทางของจัดการข้อมูลจุดยอดที่เรียกว่าเพอร์เวอร์เท็กโอเพอเรชัน (per-vertex opertations) ซึ่งจะทำหน้าที่แปลงข้อมูลจุดยอดให้กลายเป็นรูปทรง การคำนวณการแปลงภาพลายผิว การคำนวณลักษณะของแสงที่จุดยอดแต่ละจุดได้รับ รวมทั้งคำนวณการแปลงจากกรอบพิกัดอ้างอิงที่กำหนดมาให้อยู่ในกรอบอ้างอิงกล้องที่จะใช้แสดงผล การประมวลผลอีกส่วนหนึ่งเรียกว่าพริมมิทิฟแอสเซมบลี (primitive assembly) ซึ่งการประมวลผลที่สำคัญในกลุ่มนี้คือการขลิป (clipping) ส่วนของรูปทรงที่อยู่นอกขอบเขตการแสดงผลข้อมูลต่าง ๆ ที่ได้จากกระบวนการนี้จะเป็นพื้นฐานสำหรับการประมวลผลในขั้นตอนแปลงให้เป็นแรสเตอร์ (rasterization)

ข้อมูลอินพุตอีกส่วนหนึ่งคือข้อมูลพิกเซล (pixel data) ซึ่งจะเป็นข้อมูลของภาพลายผิวที่ผู้ใช้กำหนดมาหรืออาจจะเป็นข้อมูลพิกเซลของภาพในเฟรมบัฟเฟอร์ก็ได้ ข้อมูลส่วนนี้จะแยกการประมวลผลไปตามเส้นทางที่เรียกว่าเพอร์พิกเซลโอเพอเรชัน (per-pixel operations) โดยตัวดำเนินการในกลุ่มนี้จะเตรียมข้อมูลจุดภาพให้อยู่ในรูปที่สามารถนำไปใช้ในขั้นตอนอื่น ๆ ได้ เช่น การบีบหรือขยายข้อมูลพิกเซลที่ส่งมาให้มีอยู่ในรูปแบบที่กำหนด, การปรับขนาด และการประมวลผลพิกเซลภาพ เป็นต้น ผลลัพธ์ที่ได้นั้นอาจจะถูกบันทึกกลับลงหน่วยความจำสำหรับเก็บข้อมูลภาพลายผิวหรืออาจจะถูกส่งต่อไปให้ขั้นตอนการแปลงให้เป็นแรสเตอร์ก็ได้

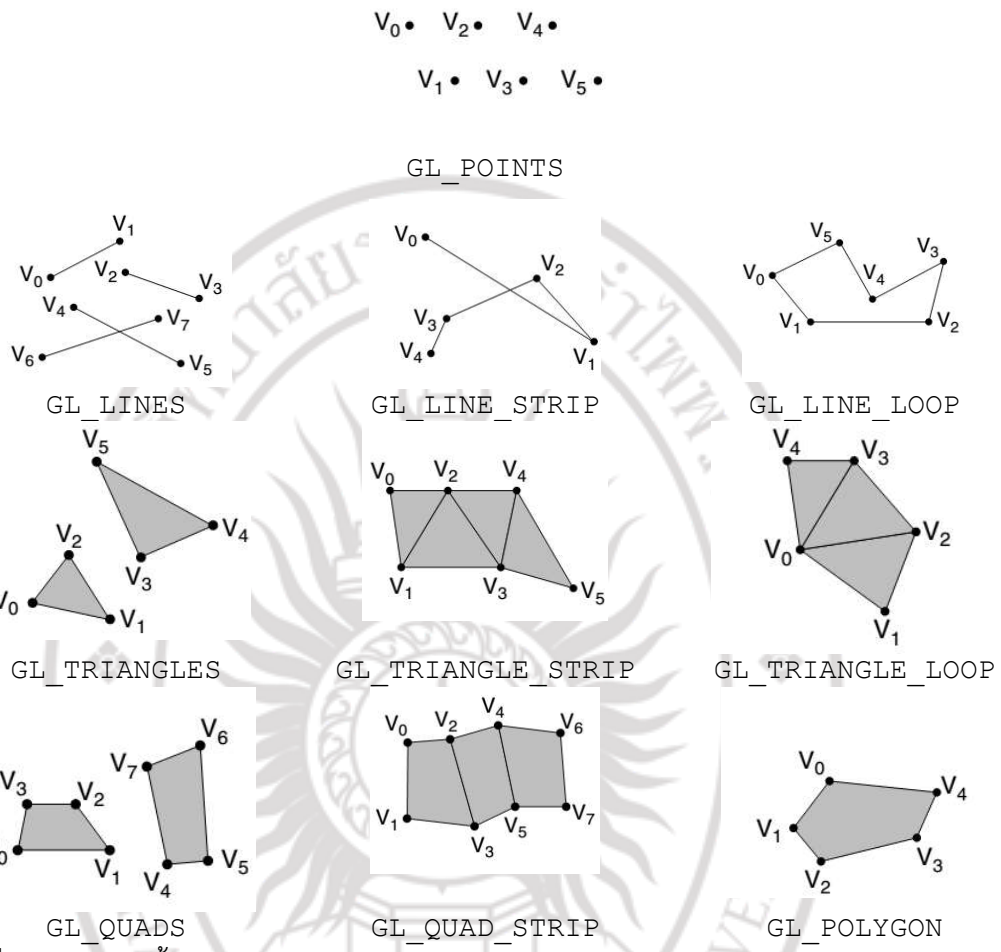
ขั้นต่อมาคือการทำให้เป็นแรสเตอร์ซึ่งจะเป็นการแปลงข้อมูลต่าง ๆ ทั้งแบบจุดยอดและแบบพิกเซลให้กระจายเป็นส่วนเล็ก ๆ ที่เรียกว่าแฟรกเมนต์ (fragment) โดยเฟรกเม้นต์แต่จะชิ้นจะเกี่ยวข้องโดยตรงกับข้อมูลสีและความลึกของพิกเซลแต่ละพิกเซลในเฟรมบัฟเฟอร์ เสร็จแล้วแฟรกเมนต์ทั้งหมดจะถูกส่งไปประมวลผลในขั้นตอนที่เรียกว่าแฟรกเมนต์โอเพอเรชัน (fragment operations) ขั้นตอนแรก ๆ ของการประมวลผลกลุ่มนี้คือการกำหนดภาพลายผิวให้แต่ละแฟรกเมนต์โดยใช้ข้อมูลพิกเซลที่ได้เตรียมไว้ในหน่วยความจำภาพลายผิว ต่อมาจึงคำนวณสีลำดับที่หนึ่ง สีลำดับที่สอง และความลึกของแต่ละแฟรกเมนต์ รวมทั้งคำนวณผลของหมอก (fog) ที่มีต่อแต่ละแฟรกเมนต์ด้วย หลังจากนั้นจะเข้าสู่การตรวจประเมินข้อมูลแฟรกเมนต์ด้วยการทดสอบแบบต่าง ๆ เช่น ซิสเซอร์เทส (scissor test) อัลฟ่าเทส (alpha test) สเทนซิลเทส (stencil test) และ เด็บบัฟเฟอร์เทส (depth-buffer test) รวมทั้งการคำนวณการผสมสีต่าง ๆ ก็จะทำในขั้นตอนนี้ด้วย โดยแฟรกเมนต์ใดที่ไม่ผ่านการประเมินก็จะถูกละทิ้งและไม่นำไปแสดงผลบนหน้าจอ และแฟรกเมนต์ที่ผ่านการทดสอบจะถูกนำไปวางไว้ในเฟรมบัฟเฟอร์เพื่อการแสดงผลต่อไป

2.3.3 โอเพนจีแอล

ในแง่ของผู้ใช้นั้นจะมองว่าเราสามารถวาดรูปทรงใด ๆ ด้วยโอเพนจีแอลก็ได้ แต่ในความเป็นจริงแล้วโอเพนจีแอลนั้นมีความสามารถในการวาดรูปทรงเพียงไม่กี่แบบเท่านั้น เราจะเรียกกลุ่มของรูปทรงที่โอเพนจีแอลรองรับเหล่านี้ว่ารูปทรงพื้นฐาน (baisc primitives) ภาพที่ 2.8 แสดงตัวอย่างของรูปทรงพื้นฐานที่สามารถวาดได้ในโอเพนจีแอล และตารางที่ 2.4 อธิบายความหมายของคำสั่งเหล่านั้นเมื่อใช้วาดจุดยอด $\mathbf{v} = \{\mathbf{v}_1, \mathbf{v}_2, \ldots \mathbf{v}_n\}$

ตารางที่ 2.4 ความหมายของคำสั่งวาดรูปทรงพื้นฐานในภาพที่ 2.8
ที่มา: Shreiner et al, 2013

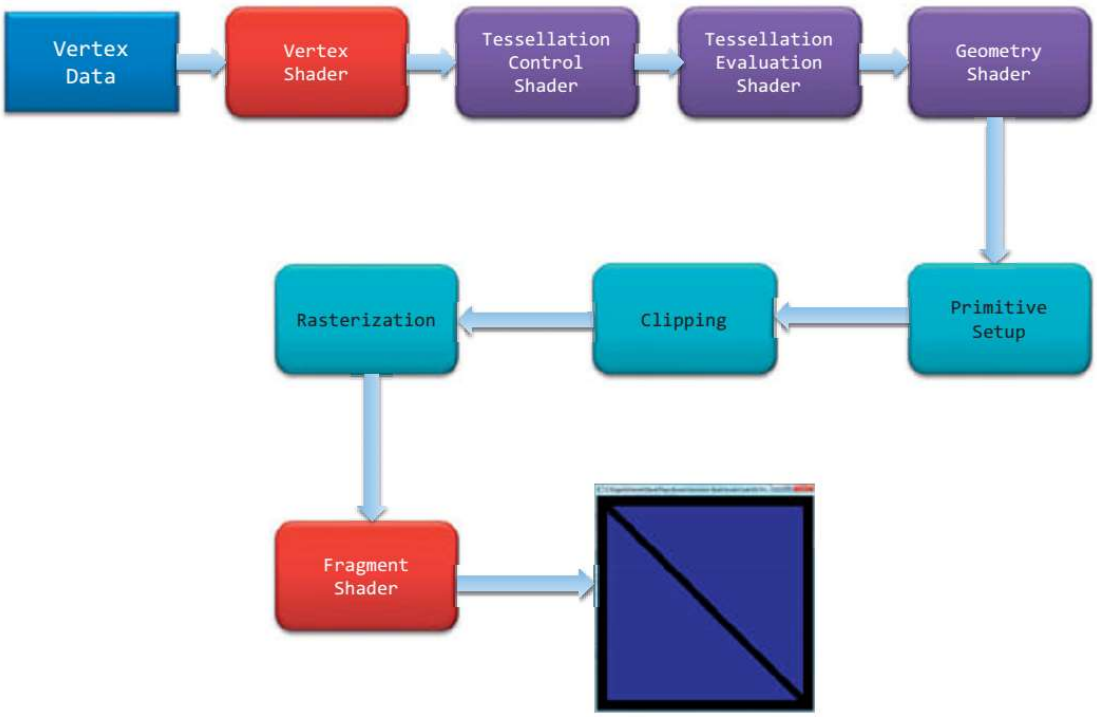
ค่าคงที่	ความหมาย
GL_POINTS	วาดจุดที่จุดยอดแต่ละจุดทุกจุด
GL_LINES	วาดชุดของส่วนของเส้นตรงที่ไม่เชื่อมต่อกัน ส่วนของเส้นตรงแต่ละเส้นจะกำหนดด้วยจุดยอดสองจุด โดยเส้นแรกจะใช้จุดยอด $\{\mathbf{v}_1, \mathbf{v}_2\}$ เส้นที่สองจะใช้ $\{\mathbf{v}_3, \mathbf{v}_4\}$ ตามลำดับไปเรื่อย ๆ และหาก n เป็นเลขคี่แล้วส่วนของเส้นตรงเส้นสุดท้ายจะถูกละทิ้งไปเนื่องจากข้อมูลจุดยอดไม่พอ
GL_LINE_STRIP	วาดชุดของส่วนของเส้นตรงที่มีจุดยอดเชื่อมต่อกัน นั่นคือเส้นแรกจะใช้จุดยอด $\{\mathbf{v}_1, \mathbf{v}_2\}$ แล้วต่อด้วย $\{\mathbf{v}_2, \mathbf{v}_3\}$ ไปตามลำดับจนถึง $\{\mathbf{v}_{n-1}, \mathbf{v}_n\}$
GL_LINE_LOOP	คล้ายกับ GL_LINE_STRIP แต่จะมีเส้นสุดท้ายเพิ่ม ลากจุด $\mathbf{v}_n$ ไปยัง $\mathbf{v}_1$ เพื่อให้ได้เป็นรูปปิด
GL_TRIANGLES	วาดชุดของรูปสามเหลี่ยมที่ไม่เชื่อมต่อกัน โดยจะใช้จุดยอดครั้งละสามจุด $\{\mathbf{v}_1, \mathbf{v}_2,\mathbf{v}_3\}$, $\{\mathbf{v}_4, \mathbf{v}_5,\mathbf{v}_6\}$... และจะไม่วาดภาพสามเหลี่ยมหากมีข้อมูลจุดยอดไม่ครบสามจุด
GL_TRIANGLE_STRIP	วาดชุดของรูปสามเหลี่ยมที่เชื่อมเส้นขอบต่อกัน โดยสามเหลี่ยมรูปแรกจะใช้จุดยอด $\{\mathbf{v}_1, \mathbf{v}_2,\mathbf{v}_3\}$ จากนั้นสามเหลี่ยมรูปที่สองจะใช้จุดยอดสองจุดสุดท้ายของจุดแรกร่วมกันเป็น $\{\mathbf{v}_2, \mathbf{v}_3,\mathbf{v}_4\}$ ตามลำดับ การใช้เส้นขอบร่วมกันลักษณะนี้จะทำให้รูปสามเหลี่ยมมีการเอียงตัวหรือมีเวกเตอร์ปกติชี้ไปในทิศทางเดียวกันทั้งหมด (ซึ่งมีความสำคัญอย่างมากกับการคำนวณการให้แสงและเงา)
GL_TRIANGLE_FAN	วาดชุดของสามเหลี่ยมที่เชื่อมต่อกันด้วยจุดยอดจุดแรก โดยสามเหลี่ยมรูปแรกจะใช้จุดยอด $\{\mathbf{v}_1, \mathbf{v}_2,\mathbf{v}_3\}$ จากนั้น $\{\mathbf{v}_1, \mathbf{v}_4,\mathbf{v}_5\}$ ตามลำดับ
GL_POLYGON	วาดโพลีกอนโดยใช้ชุดของจุดยอดที่กำหนดมา ทั้งนี้ต้องมีจุดยอดอย่างน้อยสามจุด และจุดยอดเหล่านี้จะต้องไม่มีทำให้เส้นขอบของโพลีกอนที่วาดได้ตัดกับตัวเอง และจะต้องไม่ทำให้โพลีกอนที่วาดได้เป็นรูปแบบเว้าเข้า (concave) ถ้าไม่เช่นนั้นแล้วโปรแกรมอาจจะแสดงผลไม่ถูกต้อง
GL_QUADS	วาดชุดของรูปสี่เหลี่ยม (quadrilateral) แบบไม่เชื่อมต่อกันคล้ายกับการวาดสามเหลี่ยมด้วย GL_TRIANGLES โดยแต่จะใช้จุดยอดครั้งละสี่จุด
GL_QUAD_STRIP	คล้ายกับ GL_TRIANGLE_STRIP แต่จะเป็นการวาดภาพสี่เหลี่ยมและใช้จุดยอดครั้งละสี่จุด



ภาพที่ 2.8 รูปทรงพื้นฐานของโอเพนจีแอล
ที่มา: Shreiner et al, 2013

2.3.4 แรนเดอร์ริงไปป์ไลน์แบบใหม่

การวาดวัตถุและไปปไลน์ของโอเพนจีแอลที่กล่าวมาในหัวข้อที่ผ่านมานั้นเป็นวิธีการทำงานของโอเพนจีแอลรุ่นแรก ๆ โดยใช้เรนเดอร์ริงไปป์ไลน์แบบคลาสสิก แต่ปัจจุบันปัจจุบันได้มีการปรับปรุงไปป์ไลน์นี้ให้รองรับการทำงานที่หลากหลายมากขึ้น รวมทั้งโดยเฉพาะอย่างยิ่งให้รองรับการทำงานร่วมกับจีพียูเพื่อให้โปรแกรมสามารถแสดงผลได้มีประสิทธิภาพมากขึ้น โดยเราสามารถสรุปเรนเดอร์ริงไปป์ไลน์แบบใหม่ที่ใช้ในโอเพนจีแอล 4.3 ดังแสดงในภาพที่ 2.9



ภาพที่ 2.9 เรนเดอร์ริงไปป์ไลน์ของโอเพนจีแอล 4.3
ที่มา: ดัดแปลงจาก Shreiner et al, 2013

จากภาพจะเห็นว่าข้อมูลจุดยอดนั้นจะถูกส่งผ่านไปยังกลุ่มของฟังก์ชันที่เรียกว่าเฉดเดอร์ (shader) ก่อนที่จะผ่านเข้าไปยังกระบวนการขลิปและแปลงให้เป็นแรสเตอร์ หลังจากนั้นจะผ่านเฉดเดอร์อีกหนึ่งชุดก่อนจะไปปรากฏบนเฟรมบัฟเฟอร์ เราจะเห็นว่าการดำเนินการแบบใหม่เพิ่มเข้ามาคือเฉดเดอร์ซึ่งเป็นกลุ่มของโปรแกรมขนาดเล็กที่ออกแบบให้มีการประมวลผลบางขั้นตอนบนจีพียู ดังนั้นเฉดเดอร์จึงอาจจะช่วยเพิ่มประสิทธิภาพในการแสดงผลให้ดีขึ้นกว่าการประมวลผลบนซีพียูเพียงอย่างเดียว ในโอเพนจีแอลเวอร์ชันแรกจนถึงเวอร์ชัน 3.0 นั้นผู้ใช้สามารถวาดวัตถุได้จากฟังก์ชันกลุ่มที่เรียกว่าฟิกซ์ฟังก์ชัน (fixed function) เช่น ในไปป์ไลน์แบบคลาสสิกนั้นจะส่งข้อมูลจุดยอดด้วยฟังก์ชัน `glBegin()`, `glEnd()` และ `glColor()` และจากเฉดเดอร์ผ่านการใช้งานจากส่วนขยาย ร่วมกับการส่งข้อมูลในรูปของแถวลำดับด้วยฟังก์ชันอีกกลุ่มหนึ่ง เช่น `glBufferData()` และ `glDrawArrays()` อย่างไรก็ตามหลังจากเวอร์ชัน 3.1 เป็นต้นมา ฟังก์ชันกลุ่มฟิกซ์ฟังก์ชันนั้นถูกลดความสำคัญและถูกลบออกจากไปในโอเพนจีแอลเวอร์ชั่น 4 แล้วให้เฉดเดอร์และการส่งข้อมูลแบบแถวลำดับเข้ามาทำหน้าที่แทน

เราสามารถเขียนโปรแกรมเฉดเดอร์โดยใช้จีแอลเอสแอล (OpenGL Shading Language : GLSL) ซึ่งเป็นภาษาระดับสูง (hi-level shading language) ที่มีลักษณะคล้ายภาษาซีและซีพลัสพลัส โดยก่อนหน้านี้เราจะใช้งานเฉดเดอร์ได้ในลักษณะของส่วนขยายแต่เริ่มมีการกำหนดใช้อย่างจริงจังตั้งแต่โอเพนจีแอล 2.0 เป็นต้นมา (Shreiner et al, 2013) มีเฉดเดอร์ที่สำคัญอยู่ห้าเฉดเดอร์ที่ทำหน้าที่ต่าง ๆ กัน โดยเวอร์เท็กซ์เฉดเดอร์ (vertex shader) จะทำหน้าที่ประมวลผลจุดยอดแต่ละจุดและข้อมูลประกอบอื่น ๆ ที่ผู้ใช้ส่งมาพร้อมกับจุดยอดนั้น ๆ เช่น เวกเตอร์ปกติ และ

สีของจุดยอด หน้าที่หลักอย่างหนึ่งของเฉดเดอร์นี้คือการคำนวณการแปลงพิกัดของจุดยอดจากกรอบอ้างอิงที่ผู้ใช้กำหนดมาเป็นกรอบอ้างอิงกล้องสำหรับการแสดงผล รวมไปถึงการคำนวณแสงและสีของจุดยอดด้วย นอกจากนั้นแล้วผู้ใช้ยังสามารถเขียนโปรแกรมเพื่อจัดการจุดยอดได้ด้วย เช่น การเพิ่มจุดยอดเพื่อวาดรูปทรงอื่น ๆ เพิ่มเติม และการดัดแปลงพิกัดของจุดยอดเพื่อให้ได้รูปทรงเปลี่ยนไปตามที่ต้องการ

เทสเซลเลชันคอลโทรล (tessellation control) และ อีวาลูเอชันเฉดเดอร์ (evaluation shaders) เป็นเฉดเดอร์ทางเลือก โดยหากมีการเปิดการใช้งานเฉดเดอร์นี้จะทำหน้าที่เพิ่มจำนวนรูปทรงพื้นฐาน (เช่นสามเหลี่ยมหรือโพลีกอน) ของวัตถุที่จะวาดด้วยเทคนิคเทสเซลเลชัน (tesselation) สำหรับจีโอเมตรีเฉดเดอร์ (geometry shader) เป็นเฉดเดอร์ทางเลือกเช่นกัน โดยหากมีการเปิดการใช้งานเฉดเดอร์นี้จะสามารถแก้ไขข้อมูลต่าง ๆ ของรูปทรงแต่ละรูปที่จะวาดได้โดยตรง รวมทั้งยังสามารถเพิ่มรูปทรง เปลี่ยนประเภทรูปทรง (เช่น เปลี่ยนจากจุดไปเป็นสามเหลี่ยม) หรือลบรูปทรงที่จะวาดออกไปจากไปป์ไลน์ก็ได้เช่นเดียวกัน และสุดท้ายคือแฟรกเมนต์เฉดเดอร์ (fragment shader) จะทำหน้าที่ประมวลผลแฟรกเมนต์ซึ่งโดยปกติมักจะเป็นการคำนวณสีของแต่ละแฟรกเมนต์ซึ่งผู้ใช้สามารถคำนวณสีของเฟรกเม้นต์ได้โดยการดำเนินการทางคณิตศาสตร์กับค่าสี ค่าความลึก และความเข้มของหมอก รวมถึงการละทิ้งแฟรกเมนต์บางชิ้นได้อีกด้วย

นอกจากเฉดเดอร์ทั้งสี่กลุ่มนี้แล้ว จีแอลเอสแอลยังมีเฉดเดอร์อีกหนึ่งชุดที่ไม่ปรากฏอยู่ในไปป์ไลน์ตามปกติ เฉดเดอร์นี้ชื่อว่าคอมพิวต์เฉดเดอร์ (compute shader)

2.3.5 โอเพนจีแอลอีเอส (OpenGL ES)

โอเพนจีแอลสำหรับระบบฝังตัวหรือโอเพนจีแอลอีเอส (OpenGL for Embedded Systems: OpenGL ES) เป็นคลังโปรแกรมเอพีไอที่มีพื้นฐานอยู่บนโอเพนจีแอล และได้รับการออกแบบมาให้ทำงานได้คล้ายกับโอเพนจีแอลเกือบทั้งหมด แต่มีเป้าหมายหลักอยู่ที่ระบบฝังตัวเช่น ระบบปฏิบัติการของอุปกรณ์เคลื่อนที่ต่าง ๆ นอกจากนั้นยังออกแบบมาให้รองรับการทำงานของจีพียูอีกด้วย ซึ่งผู้ออกแบบได้ออกแบบให้โอเพนจีแอลอีเอสมีฟังก์ชันต่าง ๆ คล้ายกับโอเพนจีแอลบนคอมพิวเตอร์ตั้งโต๊ะ ภาพที่ 2.6 (ข) แสดงโลโก้ของโอเพนจีแอลอีเอส

โอเพนจีแอลอีเอส 1.X นั้นจะไม่มีฟิกซ์ฟังก์ชัน ดังนั้นผู้ใช้จึงไม่สามารถสร้าง รูปทรงพื้นฐานเช่นสามเหลี่ยมหรือเส้นด้วยคำสั่ง `glBegin()` และ `glEnd()` ได้ นอกจากนั้นแล้วยังไม่รองการใช้วาดรูปทรงแบบ `GL_QUAD` และ `GL_POLYGON` ด้วย ข้อจำกัดนี้จะทำให้ผู้ใช้จะต้องสร้างวัตถุจากการใช้เวอร์เท็กซ์เฉดเดอร์เป็นหลัก ในขณะที่โอเพนจีแอล 2.0 จะมีพื้นฐานจากโอเพนจีแอล 3.X แต่ก็จะยังคงไม่มีฟิกซ์ฟังก์ชัน ดังนั้นชุดคำสั่งที่ใช้โอเพนจีแอล 2.0 จึงจะต้องพึ่งเฉดเดอร์ในการทำงาน

2.3.6 เว็บจีแอล

เว็บจีแอล (Web Graphic Library : WebGL) เป็นคลังโปรแกรมภาษาจาวาสคริปต์ (JavaScript) ที่ใช้สำหรับการสร้างภาพกราฟิกส์แบบสองมิติและสามมิติบนเว็บเบราว์เซอร์โดยไม่พึ่งส่วนต่อขยายหรือโปรแกมเสริม (plug-in) ใด ๆ โดยที่เว็บจีแอลเป็นคลังโปรแกรมที่พัฒนาอยู่บนหลักการของโอเพนจีแอลอีเอส 2.0 และจะใช้แคนวาส (canvas) ของเอชทีเอ็มแอลห้า (HTML5) เป็นอิลีเมนต์หลักในการแสดงผลกราฟิกส์ ภาพที่ 2.6 (ค) แสดงโลโก้ของเว็บจีแอล

วลาดีมีร์ วูคีเชห์วิค (Valadimir Vukicevie) ได้เริ่มต้นพัฒนาโปรแกรมชื่อว่าแคนวาส-ทรีดี (Canvas 3D) ในปี ค.ศ. 2006 ขึ้นโดยทดสอบการใช้งานกับเว็บเบราว์เซอร์โมซิลลา (Mozilla) และในปีต่อมาก็ได้มีการประกาศว่า ทั้งโมซิลล่าและโอเปลา (Opera) นั้นรองรับการทำงานของแคนวาสทรีดี ต่อมาในช่วงปี ค.ศ. 2009 ทางกลุ่มโครนอส (Khronos) ได้ก่อตั้งคณะทำงานเพื่อจัดการเว็บจีแอลโดยประกอบด้วยบุคลากรจากหลายหน่วยงาน เช่น แอปเปิล (Apple) กูเกิล (Google) โมซิลลา และ โอเปลา หลังจากนั้นจึงมีการประกาศข้อกำหนดอย่างเป็นทางการของเว็บจีแอลเวอร์ชันแรก (WebGL 1.0) ในเดือนมีนาคม ค.ศ. 2011 และข้อกำหนดของเว็บจีแอลเวอร์ชันล่าสุดคือเวอร์ชัน 2.0[2] ซึ่งได้ประกาศใช้ในปี ค.ศ. 2013 และโดยอยู่บนพื้นฐานของโอเพนจีแอลอีเอส 3.0 (Khronos, 2018)

อย่างไรก็ตามเว็บจีแอลนั้นจะไม่รองรับการทำงานจากฟังก์ชันในกลุ่มฟิกซ์ฟังก์ชันเช่นเดียวกับโอเพนจีแอลอีเอส นอกจากนั้นแล้วอาจจะมีประสิทธิภาพการแสดงผลบนเว็บเบราว์เซอร์แต่ละตัวไม่เท่ากัน อันอาจจะเนื่องมาจากการอิมพลิเมนต์คำสั่งเว็บจีแอลบนเว็บเบราว์เซอร์เหล่านั้น ซึ่งโดยส่วนใหญ่แล้วผู้พัฒนาเว็บเบราว์เซอร์ที่ทำงานบนระบบปฏิบัติการไมโครซอฟต์วินโดวส์ (Microsoft Windows) เช่น กูเกิลโครม (Google Chrome) และ โมซิลลาไฟร์ฟ็อกซ์ (Mozilla Firefox) จะใช้คลังโปรแกรม ANGLE (Almost Native Graphics Layer Engine)[3] ในการแสดงผล ซึ่งคลังโปรแกรมชุดนี้จะทำหน้าที่แปลงคำสั่งของโอเพนจีแอลให้กลายเป็นคลังโปรแกรมไดเรกเอ็กซ์ (DirectX)[4] แต่ในทางตรงกันข้ามเว็บจีแอลบนระบบปฏิบัติการอื่น ๆ ส่วนใหญ่จะแสดงผลด้วยโอเพนจีแอลโดยตรง

2.4 ความเป็นจริงเสริม

ความเป็นจริงเสริมหรือเออาร์ (augmented reality : AR) เป็นเทคโนโลยีที่รวมข้อมูลจากโลกจริง (real world) เข้ากับภาพกราฟิกส์ที่สร้างจากคอมพิวเตอร์ โดย Azuma (1997) และ Azuma et al. (2001) ได้อธิบายว่าความเป็นจริงเสริมมีคุณลักษณะที่สำคัญสามส่วน ส่วนแรกเป็นลักษณะของระบบการผสมภาพเสมือน (virtual image) เข้ากับภาพจากโลกจริง ส่วนที่สองเป็นการตรึงพิกัดข้อมูลดิจิทัลสามมิติเข้ากับโลกจริง และส่วนที่สามเป็นระบบที่มีปฏิสัมพันธ์กับผู้ใช้แบบเวลาจริงสามารถนำไปประยุกต์ได้หลายด้าน โดยมีรายละเอียดของเทคโนโลยีที่เกี่ยวข้องดังนี้

2.4.1 เทคโนโลยีที่เกี่ยวข้อง

Azuma et al. (2001) ได้สรุปเทคโนโลยีที่เกี่ยวข้องกับการแสดงผลด้านความเป็นจริงเสริมได้สามเทคโนโลยี เทคโนโลยีแรกคือจอภาพแบบมองผ่าน (see-though display) ซึ่งในช่วงต้นศตวรรษที่ 2000 นั้นเทคโนโลยีการแสดงภาพแบบนี้มีข้อจำกัดหลายประการ โดยเฉพาะอย่างยิ่งเรื่องความสว่าง ความละเอียด (resolution) องศารับภาพหรือเอสโอวี (field of view : FOV) และคอนทราสของภาพที่จะสามารถผสานเข้ากับภาพพื้นหลังจากโลกจริง นอกจากนั้นยังมีประเด็นเรื่อง

2 https://www.khronos.org/registry/webgl/specs/latest/

3 https://code.google.com/p/angleproject/

4 เอพีไอกราฟิกส์อีกตัวหนึ่งของไมโครซอฟต์ รายละเอียดเกี่ยวกับไดเรกเอ็กซ์นั้นสามารถหาอ่านเพิ่มเติมได้จากเว็บไซต์ของไมโครซอฟต์

ขนาด น้ำหนัก และราคาที่ยังเป็นข้อจำกัดเพิ่มเติมอีกด้วย โดยมีผู้ผลิตหลายรายที่มีส่วนเกี่ยวข้องกับการพัฒนาอุปกรณ์แสดงผลในกลุ่มนี้ เช่น โซนีและโอลิมปัส ซึ่งในช่วงต้นของทศวรรษที่ 21 ได้เริ่มมีการผลิตจอที่สามารถแสดงภาพแบบโปร่งใสได้ มีความสามารถแสดงภาพสี นอกจากนั้นยังนำมาสวมศีรษะได้ โดยมีความละเอียดภาพประมาณ 180 - 240 กิโลพิกเซล มีองศารับภาพประมาณ 30 องศา อย่างไรก็ตาม ข้อจำกัดของเทคโนโลยีแบบนี้ประการหนึ่งคือการจัดวางให้แนวมุมมองสายตาของผู้ใช้กับภาพบนจอนั้นอยู่ในแนวเดียวกันเพื่อลดระยะเหลื่อม (palallax) บนภาพ

สำหรับเทคโนโลยีการแสดงภาพกลุ่มที่สองคือจอภาพแบบฉาย (projection displays) ซึ่งในกลุ่มนี้จะใช้การฉายข้อมูลหรือภาพเสมือนของสิ่งต่าง ๆ ซ้อนทับลงบนวัตถุจริงโดยตรง โดยกรณีที่พื้นฐานที่สุดคือการพยายามฉายภาพของวัตถุให้อยู่ในระนาบเดียวกับวัตถุด้วยเครื่องฉายที่ติดตั้งภายในห้องปฏิบัติการ ซึ่งผู้ใช้ไม่จำเป็นต้องสวมใส่อุปกรณ์เพิ่มเติม ตัวอย่างของเทคโนโลยีกลุ่มนี้ คือเคฟ (Cave automatic virtual environment: CAVE) และเทคโนโลยีกลุ่มสุดท้ายคือเซ็นเซอร์ ซึ่งเป็นเทคนิคกลุ่มที่ไม่เกี่ยวกับการแสดงผลโดยตรง เป็นเทคโนโลยีที่มีการใช้เซ็นเซอร์ต่าง ๆ เพื่อตรวจวัดทิศทางการมองของผู้ใช้ ในกรณีนี้ผู้พัฒนาจะต้องมีระบบการติดตาม (track) ที่มีความแม่นยำสูงเพื่อติดตามทิศทางและตำแหน่งของผู้ใช้เพื่อการตรึงพิกัดให้ถูกต้อง

Schmalstieg et al. (2011) ได้เสนอแนวคิดเออาร์ 2.0 (Augmented Reality 2.0) ซึ่งเป็นการรวมความเป็นจริงเสริมเข้ากับแนวคิดเรื่องเว็บ 2.0 (Web 2.0) ดังแสดงในตารางที่ 2.5

ตารางที่ 2.5 เปรียบเทียบคุณลักษณะของเว็บ 2.0 กับเออาร์ 2.0
ที่มา : Schmalstieg et al. (2011)

เว็บ 2.0	เออาร์ 2.0
ผู้ใช้และเว็บมีจำนวนมาก	มีขนาดใหญ่ทั้งในแง่ของจำนวนผู้ใช้และพื้นที่ทำงาน
ไม่มีแนวแบ่งที่ชัดเจนระหว่างการเข้าถึงข้อมูลเฉพาะที่และข้อมูลระยะไกล	ไม่มีการแบ่งที่ชัดเจนระหว่างการแสดงผลข้อมูลเฉพาะที่และข้อมูลระยะไกล
โปรแกรมที่รันบนเว็บเบราว์เซอร์มีพฤติกรรมคล้ายโปรแกรมเฉพาะที่ มีการส่งเสริมให้ผู้ใช้สร้างปฏิสัมพันธ์กับระบบ	โปรแกรมทำงานแบบเฉพาะที่บนอุปกรณ์ซึ่งสามารถดาวน์โหลดมอดูล (module) หรือฟีเจอร์ (feature) ใหม่ ๆ จากเครื่องให้บริการได้จากระยะไกลโดยผู้ใช้ไม่จำเป็นต้องรับรู้
ผู้ใช้จำนวนมากไม่ใช่ผู้เชี่ยวชาญทางเทคนิคซึ่งสามารถเข้าถึงข้อมูล มีส่วนร่วมและสามารถแก้ไขข้อมูลได้	ผู้ใช้สามารถสร้างหรือปรับปรุงเนื้อหาของความเป็นจริงเสริมได้ตามตำแหน่งที่จำเพาะเจาะจง
สารสนเทศจากแหล่งข้อมูลที่หลากหลายสามารถนำมารวมเข้าด้วยกันเพื่อสร้างเป็นการประยุกต์ใหม่ที่มีมูลค่าเพิ่ม (value-added) ได้	สามารถเข้าถึงข้อมูลจากแหล่งข้อมูลเช่นบริการเว็บและนำมารวมเข้ากับเนื้อหาของความเป็นจริงเสริมได้ในปริภูมิสามมิติ

2.4.2 การตรึงพิกัด

ระบบความเป็นจริงเสริมส่วนใหญ่จะใช้จีพีเอสเพื่อระบุตำแหน่งของผู้ใช้ โดยนำพิกัดที่ได้จากเครื่องรับสัญญาณจีพีเอสมาใช้งาน อย่างไรก็ตามพิกัดที่ได้จากเครื่องรับสัญญาณจีพีเอสแบบพลเรือนนั้นมีความคลาดเคลื่อนอย่างน้อย 5 - 10 เมตร จึงทำให้การประยุกต์กับสภาพแวดล้อมเสมือนภายนอกในระยะใกล้ เช่น เมือง นั้นทำได้ยากและการประยุกต์ภายในอาคารนั้นแทบเป็นไปไม่ได้ (Schmalstieg, 2001, pp,31) นอกจากนั้นการจะซ้อนทับภาพกราฟิกส์กับภาพจากโลกจริงได้นั้นจะต้องทราบทิศทางของอุปกรณ์เพิ่มเติม ดังนั้นผู้พัฒนาส่วนใหญ่มักจะมีการใช้เข็มทิศอิเล็กทรอนิกส์ (electronic compass) และเซ็นเซอร์เฉื่อย (inertial sensors) ที่ปัจจุบันมักจะมีติดตั้งมาแล้วในสมาร์ทโฟนโดยนำมาใช้เพื่อการคำนวณการเอียงตัว โดยหน่วยวัดเฉื่อยหรือไอเอ็มยู (inertial measurement unit : IMU) เป็นอุปกรณ์อิเล็กทรอนิกส์ที่สำคัญส่วนหนึ่งในระบบนำทาง มีองค์ประกอบภายในที่สำคัญคือเซ็นเซอร์ที่วัดความเร่งหรือแรงที่กระทำกับอุปกรณ์และความเร็วเชิงมุม นอกจากนั้นอาจจะมีการรวมเข็มทิศอิเล็กทรอนิกส์ เครื่องวัดสนามแม่เหล็กโลก และอุปกรณ์อื่น ๆ เพิ่ม

อย่างไรก็ตามหน่วยวัดเฉื่อยที่พัฒนาจากอุปกรณ์ในอุปกรณ์เคลื่อนที่นั้นมีสัญญาณรบกวน (noise) มาก มีรูปแบบข้อมูลต่างกัน และ มีความถี่ของการส่งข้อมูลต่างกัน ดังนั้นจึงต้องมีเทคนิคการนำข้อมูลจากเซ็นเซอร์ต่าง ๆ เหล่านี้มารวมกันให้เกิดประโยชน์ เทคนิคการรวมข้อมูลเซ็นเซอร์เพื่อการนำทางนี้เรียกว่าการรวมสัญญาณเซ็นเซอร์ (sensor fusion) โดย Blum, Greencorn and Coperstock (2012) ได้ศึกษาความน่าเชื่อถือของการใช้ข้อมูลจากเซ็นเซอร์ในสมาร์ทโฟนเพื่อการตรึงพิกัด โดยการใช้สมาร์ทโฟนทั้งแบบที่ใช้ระบบปฏิบัติการไอโอเอส (iOS) และแอนดรอยด์ ผลการศึกษาพบว่าความคลาดเคลื่อนมีค่าเฉลี่ยประมาณ 10 - 30 เมตร ขึ้นอยู่กับอาคารและสภาพแวดล้อม ในขณะที่คลาดเคลื่อนของเข็มทิศนั้นมีค่าเฉลี่ยประมาณ 10 องศา และบางกรณีมีค่าสูงถึง 30 องศา และพบว่าความคลาดเคลื่อนสะสมของไจโรสโคป (gyroscope) นั้นมีสูงมากกว่า 4 องศาต่อวินาที

ดังนั้นปัญหาอีกประการหนึ่งของการรวมสัญญาณเซ็นเซอร์คือความคลาดเคลื่อนสะสม (cumulative error) หรือความเบี่ยงเบน (bias) ซึ่งเกิดจากอัลกอริทึมที่ใช้ในการคำนวณและเนื่องจากสัญญาณรบกวนต่าง ๆ ภายในระบบ ดังนั้นจึงมีการนำเทคนิคด้านอื่น เช่น คอมพิวเตอร์วิชัน (computer vision : CV) และการประมวลผลภาพดิจิทัล (digital image processing : DIP) เข้ามาช่วยจัดการปัญหานี้ ยกตัวอย่างเช่น You, Neumann and Azuma (1999) ได้เสนอเทคนิคการรวมข้อมูลจากเซ็นเซอร์และจากวิเคราะห์ภาพเพื่อการตรึงพิกัด โดยงานวิจัยนี้มีการแก้ไขความคลาดเคลื่อนเลื่อน (drift) ด้วยการนำพิกัดของฟีเจอร์ (feature) หรือสิ่งที่สนใจ เช่น จุดมุมหรือเส้นในเฟรมก่อนหน้ามาการประมาณค่าหาตำแหน่งฟีเจอร์ในเฟรมปัจจุบัน ซึ่งการตำแหน่งของฟีเจอร์ที่ประมาณได้นี้จะถูกนำไปเปรียบเทียบกับตำแหน่งจริงที่ได้จากการค้นหาเฉพาะที่ (local search) ด้วยเทคนิคออปติคัลโฟลว์ (optical flow) อีกครั้ง ผลการศึกษาพบว่าการใช้ข้อมูลภาพเข้ามาช่วยทำให้ความคลาดเคลื่อนลดลงอย่างชัดเจน เทคนิคที่เป็นที่รู้จักเทคนิคหนึ่งในด้านการสกัดฟีเจอร์คือซิฟท์ (Scale Invariant Feature Transform : SIFT) (Lowe, 2004) ซึ่งจะมีการใช้ตัวกรองแบบเกาส์เซี่ยน (Gaussian filter) สร้างภาพพิรามิดผลต่างของเกาส์เซียน (Difference of Gaussian : DoG)

ของภาพที่สเกลต่าง ๆ กัน เรียกฟีเจอร์ที่สกัดได้ว่าคีย์พ้อยต์ (keypoint) โดยจะสกัดได้จากค่าสุดขีดเฉพาะที่ (local extrema) ในพิรามิดผลต่างของเกาส์เซียน หลังจากนั้นจะกรองคีย์พ้อยต์ที่มีค่าความเปรียบต่างต่ำและคีย์พ้อยต์ที่อยู่ใกล้กับเส้นขอบทิ้งไป และเพื่อให้คีย์พ้อยต์ที่สกัดได้นั้นทนทานต่อการหมุน ขั้นตอนต่อไปคือการกำหนดค่าการหมุนขอคีย์พ้อยต์จากเกรเดียนต์ (gradient) พร้อมกับคำนวณตัวบอกรายละเอียด (descriptor) ของคีย์พ้อยต์นั้น โดยตัวบอกรายละเอียดนี้จะประกอบด้วยข้อมูลการเอียงตัวที่สเกลต่าง ๆ โดยคีย์พ้อยต์ที่ได้นั้นมีความทนทานต่อการเปลี่ยนแปลงความเข้มพิกเซลหรือการเปลี่ยนแปลงมุมมองภายใต้การแปลงสัมพรรค (affine transformation) การจับคู่ฟีเจอร์ด้วยวิธีเชิงพื้นที่นั้นจะเป็นการทำงานที่ใช้เวลาในการประมวลผลมาก อีกทั้งยังไม่ทนทานต่อการเปลี่ยนแปลงมุมมองอีกด้วย ดังนั้นในกรณีของซิฟท์จึงทำการจับคู่คีย์พ้อยต์ด้วยการเปรียบเทียบข้อมูลของตัวบอกรายละเอียดของคีย์พ้อยต์ด้วยการแบ่งคีย์พ้อยต์ในภาพเป็นโครงสร้างข้อมูลต้นไม้เค-ดี (k-d tree) แล้วใช้วิธีย่านใกล้เคียงใกล้สุดโดยประมาณหรือเอเอ็นเอ็น (approximate nearest neighbour : ANN) ร่วมกันในการพิจารณาความเหมือนกันของฮิสโทแกรมของเกรเดียนต์ของคีย์พ้อยต์เหล่านั้น ซึ่งอกจากซิฟท์แล้วยังมีตัวตรวจหาฟีเจอร์อื่นที่สามารถนำมาใช้ได้ ยกตัวอย่างเช่น ฟาสต์ (features from accelerated segment test : FAST) (Rosten and Drummond, 2006) ซึ่งเป็นตัวตรวจหามุมที่มีหลักการทำงานพื้นฐานคล้ายกับซิฟท์แต่ฟาสต์จะใช้การคำนวณหามุมหรือฟีเจอร์ด้วยการพิจารณาพิกเซลในวงกลมที่มีรัศมี 16 พิกเซล แต่ละพิกเซลภายในวงกลมนี้จะมีป้ายกำกับด้วยเลขจำนวนเต็ม แล้วใช้เทคนิคการเปรียบเทียบแบบเร็วเพื่อตรวจสอบว่าพิกเซลใดในกลุ่มที่มีความน่าจะเป็นมุมในภาพ

Bleser and Stricker (2008) ได้นำเสนอเทคนิคการรวมสัญญาณโดยการใช้ทั้งข้อมูลจากหน่วยวัดเฉื่อย และข้อมูลจากการประมวลผลภาพ โดยข้อมูลจากหน่วยวัดเฉื่อยจะถูกใช้เพื่อการประมาณค่าตำแหน่งและทิศทางโดยประมาณ จากนั้นจะมีการปรับแก้ด้วยเทคนิคคาลมานฟิลเตอร์แบบขยายหรืออีเคเอฟ (extended kalman filter : EKF) ข้อมูลที่ได้นี้จะถูกนำไปใช้ในการเรนเดอร์แบบจำลองสามมิติ จากนั้นจึงสกัดฟีเจอร์ในภาพโดยใช้เทคนิคฟาสต์ หลังจากนั้นจึงเปรียบเทียบฟีเจอร์บนภาพที่เรนเดอร์ได้เทียบกับภาพจริงเพื่อปรับแก้ความคาดเคลื่อนของการประมาณค่าตำแหน่งและทิศทางอีกครั้ง ผลการศึกษาพบว่าเมื่อมีการเคลื่อนกล้องมากจะทำให้เกิดความคลาดเคลื่อนของการตรึงพิกัดมากกว่าการเคลื่อนไหวที่ช้า โดยมีค่าเฉลี่ยความคลาดเคลื่อนของการเคลื่อนกล้องเร็วอยู่ที่ประมาณ 3 พิกเซลในขณะที่ความคลาดเคลื่อนของการเคลื่อนกล้องช้านั้นมีความคลาดเคลื่อนประมาณ 0.5 พิกเซล และยังมีการสรุปว่าการใช้เซ็นเซอร์วัดความเร่ง (accelerometer) นั้นมีความแม่นยำมากกว่าการใช้ไจโรสโคปอีกด้วย

2.4.3 คลังโปรแกรมด้านเออาร์

วูโฟเลีย (Vuforia) เป็นชุดเครื่องมือพัฒนาความเป็นจริงเสริมแบบเชิงพาณิชย์ที่เปิดความสามารถบางส่วนให้โปรแกรมเมอร์ใช้งานได้ฟรีแต่มีข้อจำกัดด้านการนำไปใช้งานเชิงพานิชย์และจำนวนอุปกรณ์ที่นำโปรแกรมไปติดตั้ง ซึ่งวูโฟเลียคำนวณตำแหน่งและทิศทางของอุปกรณ์ด้วยการใช้เทคนิคการประมวลผลภาพและคอมพิวเตอร์วิชันในการสกัดและติดตามมาร์คเกอร์ (marker) ซึ่งมาร์คเกอร์นี้สามารถเป็นได้ทั้งภาพและวัตถุสามมิติ มีความสามารถในการเรนเดอร์ได้แบบเวลาจริง

(real-time) รองรับการพัฒนาด้วยภาษาซี ซีพลัสพลัส จาวา และออปเจกทิฟซีพลัสพลัส (objective-C++) รวมถึงกลุ่มภาษาดอทเน็ต (.NET) อีกด้วย

คลังโปรแกรมด้านความเป็นจริงเสริมอีกตัวที่นิยมใช้คือเออาร์ทูลคิท (ARToolKit) ซึ่งเป็นคลังโปรแกรมแบบโอเพนซอร์สที่ใช้เทคนิคการประมวลผลภาพเข้ามาช่วยในการคำนวณตำแหน่งและทิศทางของกล้องเช่นเดียวกัน โดยเออาร์ทูลคิทถูกพัฒนาโดยกลุ่มนักวิจัยห้องปฏิบัติการเอชไอที (Human Interface Technology : HIT) ของมหาวิทยาลัยวอชิงตัน (University of Washington) ถึงแม้ว่าเออาร์ทูลคิทจะเป็นคลังโปรแกรมแบบ โอเพนซอร์ส แต่การเข้าถึงความสามารถบางอย่างนั้นจะถูกจำกัดไว้เฉพาะสิทธิ์การใช้งานแบบมืออาชีพ (professional license) เท่านั้น อย่างไรก็ตามในปี ค.ศ. 2015 บริษัท DAQRI ซึ่งเป็นบริษัทด้านความเป็นจริงเสริมได้เข้าซื้อกิจการเออาร์ทูลคิท ทำให้ในปัจจุบันมีการเผยแพร่ชุดคำสั่งทั้งหมดของเออาร์ทูลคิทรวมไปถึงส่วนที่เคยถูกจำกัดไว้ก่อนหน้านี้ด้วย ตารางที่ 2.6 แสดงตัวอย่างคลังโปรแกรมสำหรับการพัฒนาโปรแกรมประยุกต์ด้านความเป็นจริงเสริมที่น่าสนใจ

ตารางที่ 2.6 คลังโปรแกรมสำหรับการพัฒนาโปรแกรมประยุกต์ความเป็นจริงเสริม

ชื่อ	ประเภท	ปีที่เปิดตัว (ค.ศ.)	ผู้ก่อตั้ง	เจ้าของปัจจุบัน
Vuforia	ฟรีแวร์, เชิงพาณิชย์	2011	Qualcomm	PTC
AR ToolKit	โอเพนซอร์ส	1999	HIT	DAQRI
ARCore	โอเพนซอร์ส (บางส่วน)	2018	Google	Google
ARKit	โอเพนซอร์ส (บางส่วน)	2017	Apple	Apple