University Of Damascus

Faculty of Information Technology

Specialization-Artificial Intelligence

Fifth year

SMART CAR

Ahmad Abdul Karim

Mohammed Alkhouja

Omar Khader

الفهرس:

1- لماذا؟: 3
2- الهدف من المشروع: 3
3- مقدمة عن المشروع: 3
4- لمحة عامة عن النظام: 4
5- الدراسة النظرية: 4
5-1- تمهيد: 4
5-2- الرؤية الحاسوبية باستخدام نموذج الرؤية المجسمة (Stereo Vision Model): 4
5-3- حساب صورة التفاوت Calculate Disparity Image: 7
5-3-1-مطابقة خطوط المسح Matching Scanlines: 7
5-3-2- تابع الكلفة Cost Function: 9
5-3-3- الشروط على عملية المطابقة بين البكسلات Hard Constraints: 9
5-3-4-خوارزمية المطابقة The Matching Algorithm: 11
5-3-5- أمثلة على حساب صورة التفاوت (Disparity Image): 14
5-4- حساب صورة العمق: 15
5-5- بناء الخريطة Map Building: 16
5-5-1- بناء الخريطة الصغيرة Minimap Building: 16
5-5-2- دمج الخريطة الصغيرة مع الخريطة الكلية للبيئة: 17
5-6- بناء خريطة البيئة المحيطة من أجل عملية التجوالRepresentation for Navigation: 18
5-7- تمثيل الخريطة من أجل تحيد مسار الروبوت Plan Representation: 19
5-7-1- كيفية حساب الـ Cost Grid: 20
6- التصميم الأولي: 22
6-1- الجزء العتادي Hardware Part: 22
6-2- الجزء البرمجي Software Part: 22
7- التصميم التفصيلي: 26
7-1- الجزء العتادي Hardware Part: 26
7-1-1- بنية الروبوت: 26
7-1-2- ميكانيكية العمل: 26
7-2- الجزء البرمجي Software Part: 27
7-2-1- المستوى الأول من مخطط الحالات(العمليات): 27
7-2-2- المستوي الثاني من مخطط الحالات(العمليات): 30
7-2-3- بنية النظام System Structure: 34

7-2-4- المخطط التفصيلي لقلب النظام: 35
8- التحقيق Implementation: 37
8-1- الجزء العتادي Hardware Part: 37
8-2- الجزء البرمجي Software Implementation: 38
9- الاختبارات والنتائج: 48
10- الآفاق المستقبلية: 54
10-1- التحسينات: 54
10-2- التطويرات: 54
11- المراجع: 55

1- لماذا؟:

هناك العديد من الأسباب التي دفعتنا إلى اختيار هذا المشروع والعمل به والتي يمكن تلخيصها فيما يلي:

1- مشروع جديد من نوعه مختلف عن المشاريع التي قمنا بتنفيذها.
2- يشمل المشروع العديد من الأفكار في مجالات مختلفة.
3- يجمع بين مختلف المواد التي قمنا بدراستها في الجامعة (الروبوتية، الرؤية الحاسوبية،هندسة البرمجيات).
4- يحتوي المشروع على جزء عتادي (Hardware) بالإضافة إلى جزء برمجي (Software).
5- إمكانية تطبيقه في الحياة العملية والعمل على تطويره في المستقبل.

2- الهدف من المشروع:

بناء روبوت متنقل (Mobile Robot) يعتمد على الرؤية الحاسوبية من أجل اكتشاف بيئة غير معروفة بالنسبة له والتجوال فيها بشكل مستقل و آمن ويقوم ببناء خريطة لهذه البيئة، واستخدام الخريطة المبنية أثناء التنقل ضمن البيئة فيما بعد دون الحاجة إلى بنائها مرة أخرى.

3- مقدمة عن المشروع:

الروبوتات المتنقلة (Mobile Robots) هي روبوتات لها القدرة على التحرك والتجوال ضمن بيئتها وهي ليست مثبتة بموقع فيزيائي محدد، يمكن التمييز بين عدة أنواع من الروبوتات المتنقلة من حيث الطريقة التي تقوم بالتجوال ضمن بيئتها فمنها من يتم التحكم بها عن طريق آلة تحكم ولا تستطيع من تلقاء نفسها التجوال، أما الروبوتات من النوع الثاني فهي التي لديها القدرة على التجوال واكتشاف الحواجز من تلقاء نفسها ويطلق عليها اسم الروبوتات المستقلة (Autonomous Robots) وهو النوع الذي سوف نحاول تقديمه وتوصيفه في مشروعنا هذا**[1]**.

الروبوتات المستقلة هي روبوتات تقوم بتأدية مهمات معينة في بيئة غير معروفة بالنسبة لها دون الحاجة إلى توجيه دائم من قبل الإنسان. تختلف درجة الاستقلالية من روبوت لآخر، حيث نجد أعلى درجات الاستقلالية مطلوبة في المهمـــات التي تتعلق باكتشـــاف الفضـاء نظرا لوجود تأخير (Delay) في الاتصال بين الأرض والفضاء بالإضافة إلى إمكانية حدوث انقطاعات (Disconnection) أثناء عملية التوجيه. يمكن الوصول إلى هذه الدرجة من الاستقلالية عند الروبوتات عن طريق تزويدها بالإدراك (Perception).

يعتبر الإدراك (Perception) من الأجزاء الهامة جداً والواجب توافرها في تصميم الروبوت المتنقل (Mobile Robot) وخاصة في حـــال حـــاجتنا لعمل الروبوت في بيئة غير معروفة بالنسبة إليه (Unknown & Unstructured Environment). للوصول لهذا الهدف لابد من تزويد الروبوت بالقدرة على إدراك البيئة المحيطة بشكل فعال لكي يستطيع العمل فيها بشكل آمن.

من المعروف أنّ معظم الروبوتات المتنقلة التي تقوم بالتجوال (Navigation) ضمن بيئة غير معروفة بالنسبة لها تعتمد بشكل أساسي على تقنية السونار(Sonar Transducers) أو على حساسات الليزر (Laser Range Sensors) من أجل اكتشاف العوائق (Obstacle Detection) وبناء الخرائط لهذه البيئة ونادرا ما تستخدم الرؤية الحاسوبية لهذه الأغراض.

في هذا المشروع سوف نقوم بتقديم روبوت متنقل يقوم باستخدام الرؤية الحاسوبية من أجل القيام بعملية التجوال ضمن بيئة غير معروفة بشكل آمن ويقوم ببناء خريطة لهذه البيئة. تكمن المشكلة في هذا الروبوت في الأخطاء التي

تظهر عند القيام بالتحويل من فضاء 3D إلى فضاء 2D بالإضافة إلى بعض الأخطاء الناتجة عن القياسات أثناء حركة الروبوت، والتي سوف نقوم بالحد من تأثيرها على عمل الروبوت وبالتالي زيادة وثوقيته (Reliability).

4- لمحة عامة عن النظام:

ينقسم النظام إلى جزأين أساسيين هما: الجزء العتادي والجزء البرمجي، وبالتالي سيتم تقسيم الشرح في هذا التقرير على هذين الجزأين. فيزيائيا يتكون النظام من روبوت متنقل على شكل سيارة يتم التحكم به من قبل برنامج منصب على حاسوب مع وجود طريقة للتخاطب بين البرنامج والروبوت المتنقل وذلك لجمع معلومات عن الروبوت (مثل الموقع وحالة الروبوت ومعلومات عن البيئة المحيطة- باستخدام آلتي تصوير) أو إعطاء أوامر له.

5- الدراسة النظرية:

5-1- تمهيد:

تهدف الدراسة النظرية إلى طرح الأفكار العلمية المستخدمة في المشروع ومناقشتها، بالإضافة إلى الطرق المستخدمة من أجل تطبيقها. يحتوي المشروع على الأفكار الرئيسية التالية:

1- إدراك البيئة المحيطة بالروبوت وذلك عن طريق استخدام الرؤية الحاسوبية.
2- بناء خريطة للبيئة المحيطة عن طريق ترجمة الإدراك.
3- حركة الروبوت في بيئة تحتوي على عوائق.

تمثل هذا الأفكار الأهداف الرئيسية التي نسعى إلى تطبيقها في هذا المشروع.

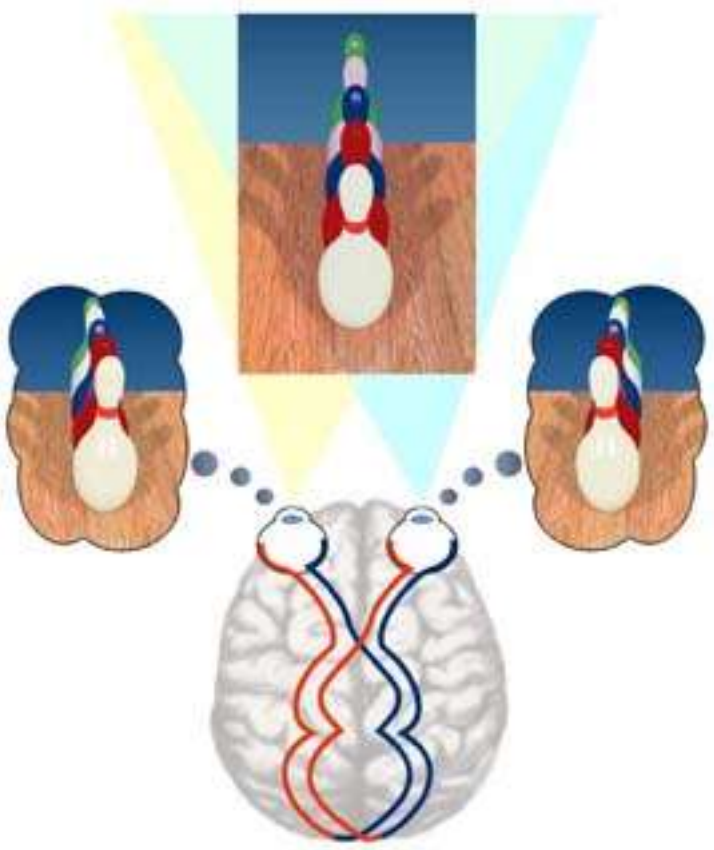
5-2- الرؤية الحاسوبية باستخدام نموذج الرؤية المجسمة (Stereo Vision Model):

من المعروف أن الإنسان يستطيع الرؤية بشكل ثلاثي الأبعاد نظراً لامتلاكه لعينين، حيث يتم الحصول على صورتين للمشهد، صورة من العين اليمنى وأخرى من العين اليسرى ومن ثم يتم الدمج بين هاتين الصورتين في الدماغ للحصول على صورة ثلاثية الأبعاد التي يستطيع الإنسان عن طريقها إدراك العمق في الصورة (انظر الشكل 1). إن إدراك البعد الثالث عند الإنسان يعتمد بشكل عام على ثلاث ظواهر هي **[6]**:

1- الظاهرة الأولى:نظام الرؤية في العين يجب أن نركز على النقطة التي ننظر عليها مما يعطينا القدرة على تحديد مسافة هذه النقطة ولكن عادة لا يتم استخدام هذه الظاهرة من قبل الإنسان وإنما يتم استخدامها في التصوير الضوئي (Photography).
2- الظاهرة الثانية: وهي الأهم هي وجود عينين عند الإنسان، والتي تسمح لنا برؤية صورتين مختلفتين من نقطتين مختلفتين، وبتحليل الفروق بين هاتين الصورتين يستطيع الإنسان إدراك بعض الخصائص الثلاثية الأبعاد للعالم المحيط.
3- الظاهرة الثالثة: الارتباط المعقد بين النظام البصري والعقل، حيث يمكن استخدام عين واحدة (إدراك الخصائص الثلاثية الأبعاد للعالم المحيط وذلك عن طريق المعرفة المكتسبة).

نظراً لعدم القدرة على تمثيل الدماغ في عمليات الرؤية الحاسوبية بحيث يمكننا رؤية الأغراض بشكل ثلاثي أبعاد من صورة واحدة، فسوف نركز في عملنا على الظاهرة الثانية والتي تعتمد على وجود صورتين والذي يطلق عليه اسم Stereo Vision، **[9] [8] [7]**.

يطلق عادة على الرؤية المجسمة Stereo Vision اسم Stereopsis **[5]** وهي مؤلفة من كلمتين Stereo وهي تعني Solidity و Opsis والتي تعني Vision. تعرف الرؤية المجسمة Stereo Vision على أنها عملية الإدراك البصري التي تؤدي إلى إدراك العمق التجسيمي (Stereoscopic Depth)، والعمق التجسيمي هو الإحساس الذي ينشأ من اندماج الإسقاطين المختلفين للعالم على شبكيتي العين. الاختلاف بين الصورتين والناتج عن المسافة الأفقية بين العينين يطلق عليه اسم التفاوت المتعلق بالعينين (Binocular Disparity) أو التفاوت المتعلق بالشبكيتان (Retinal Disparity). أول من اكتشف أن التفاوت المتعلق بالعينين يفسر من قبل الدماغ على أنه عمق الأغراض في المشهد هو العالم الإنجليزي تشارلز ويطستون Charles Wheatstone والتي نشرها في وثيقة نشرت في عام 1838، من أجل إثبات أفكاره قام تشارلز ويطستون باختراع آلة أطلق عليها اسم Stereoscope.

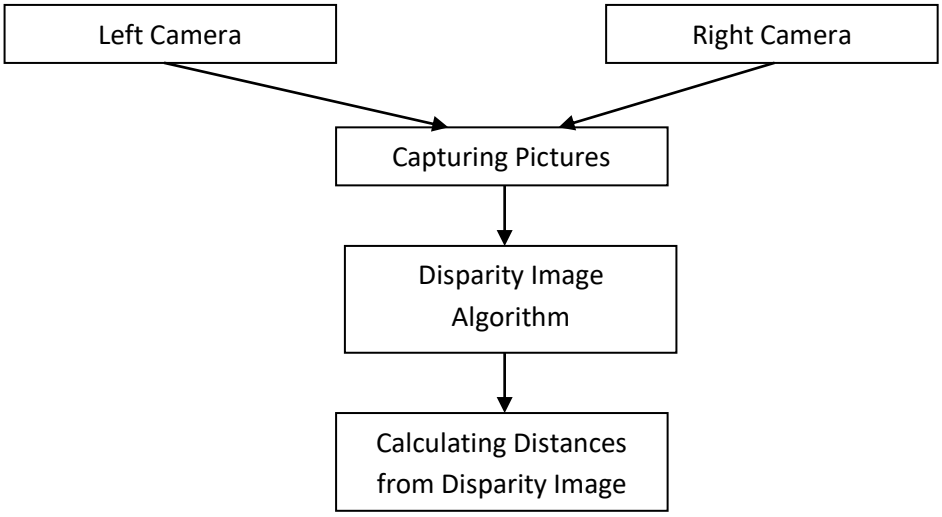


الشكل(1): توضيح لكيفية الرؤية عند الإنسان

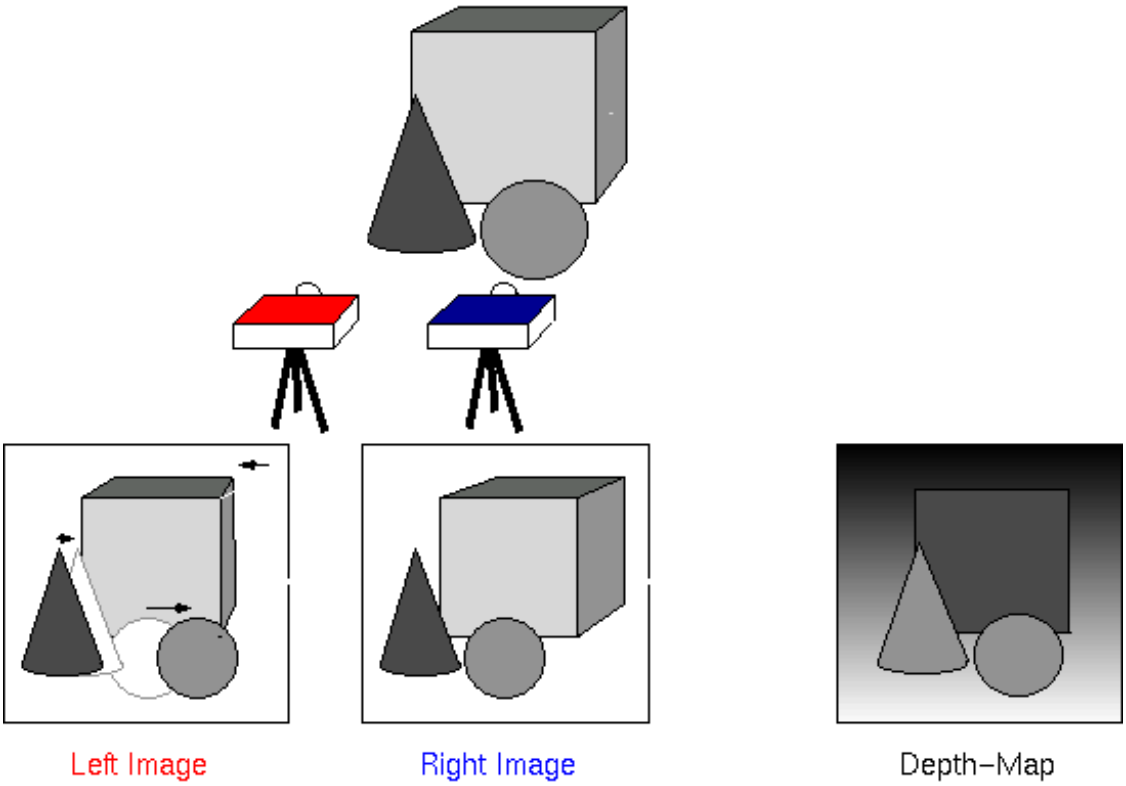
تعتبر الرؤية المجسمة (Stereo Vision) جزء من حقل الرؤية الحاسوبية (Computer Vision). تعتمد الرؤية المجسمة على مبدأ الرؤية عند الإنسان من أجل الحصول على صورة ثلاثية الأبعاد، حيث يتم الاستعاضة عن العينين عند الإنسان بآلتي تصوير تبعدان عن بعضهما مسافة معينة ويتم أخذ صورتين للمشهد ومن هاتين الصورتين يتم بناء صورة تمثل بعد الأغراض في المشهد عن آلتي التصوير يطلق على هذه الصورة الناتجة اسم Disparity Image، عن طريق خوارزمية معينة يطلق عليها اسم Disparity Image Algorithm (انظر الشكل 2). تمثل هذه الخوارزمية دور الدماغ في بناء نموذج ثلاثي أبعاد للمشهد من الصورتين اللتين نحصل عليهما من آلتي التصوير. بالنسبة للإنسان تقوم العينان بتغيير زاويتهما وذلك حسب الغرض الذي يتم رؤيته والتي تضيف بعض التعقيدات إلى الحسابات في الحاسوب، وبالتالي فإن الحالة الهندسية الأبسط هي عندما تكون مستويات صورتي الكاميرا تقع ضمن نفس المستوي. يمكن أن نقوم بعملية إعادة الإسقاط (Reprojection) من خلال تحويل خطي (Linear

Transformation) لكي تصبح على نفس المستوي، يطلق على هذه العملية اسم تقويم الصورة (Image Rectification). في بعض الأحيان يمكن أن نقوم بعملية الرؤية المجسمة (Stereo Vision) باستخدام كاميرا واحدة ويطلق عليها اسم Structure From Motion.

يمثل الشكل التالي طريقة عمل النموذج المؤلف من آلتي تصوير:



الشكل(2): المخطط العام لعمل نموذج الرؤية المجسمة



الشكل(3): الحصول على صورة التفاوت من صورتين لنفس المشهد

3-5- حساب صورة التفاوت Calculate Disparity Image:

يوجد العديد من الخوارزميات التي تقوم بحساب Disparity Image، أحد هذه الخوارزميات يطلق عليها اسم Disparity Image by Pixel-to-Pixel Stereo، تعرف هذا الخوارزمية بهذا الاسم نظرا لأنها تقوم بمقارنة الصور على مستوى البكسلات**[2]**.

تهدف هذه الخوارزمية إلى حساب الـ Disparity Image من صورتين مأخوذتين من آلتي تصوير لنفس المشهد و تبعدان عن بعضهما مسافة معينة، من أجل تحقيق هذا الهدف تقوم هذه الخوارزمية بمطابقة البكسلات بشكل فردي ضمن خطوط المسح المتناظرة في كل من الصورتين، وإبقاء بعض البكسلات (Occluded Pixels) غير متطابقة، ومن ثم تقوم بنشر المعلومات بين خطوط المسح باستخدام طريقة معالجة سريعة (Fast Postprocessor)، تستطيع هذه الخوارزمية معالجة مناطق غير منقشة كبيرة (large untextured regions)و تستخدم مقياس لحساب عدم التشابه بين البكسلات الذي لا يتأثر بتقطيع الصورة (Image Sampling) وتقوم بتشذيب العقد التي لا تفيد في البحث من أجل زيادة السرعة في البرمجة الديناميكية، إن الحسابات في هذه الخوارزمية هي سريعة نسبيا، حيث تستغرق حوالي 600 نانو ثانية لكل بكسل من الـ Disparity Imageعلى معالج تردد 800 MHz.

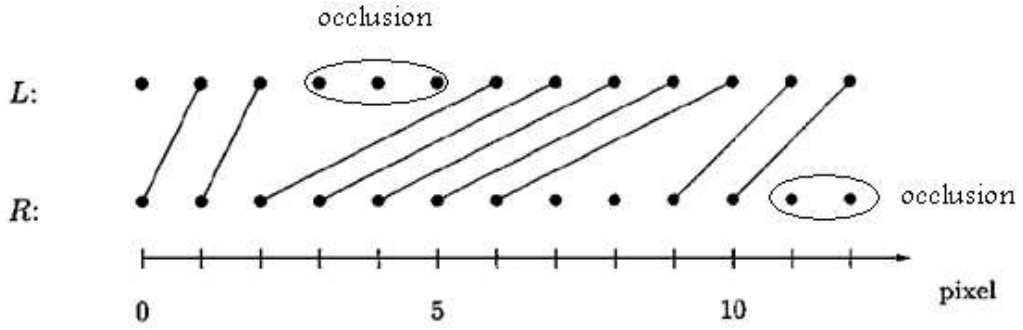
لقد عرف العاملون في مجال أفلام الكرتون أهمية عمق الصورة (Depth Discontinuities)، وبالتالي من أجل خلق الشعور بالعمق في الصورة يتم رسم الشخصيات والخلفية على طبقات مختلفة آخذين بالاعتبار الرسم الواضح للأغراض التي لا تقع في الخلفية، بشكل مشابه في الرؤية عند الإنسان فإنه يتم إدراك صورة العمق واستخدامها من أجل الحصول على أغراض منفصلة ويقوم بتحديد المسافة بين هذه الأغراض. تلعب صورة العمق دورا أساسيا في فهم الصورة بالإضافة إلى استخدامها في تقطيع الصورة نظرا لقدرته على الحصول على أغراض منفصلة.

في هذا الجزء من التقرير سوف نقوم بشرح طريقة من أجل حساب صورة التفاوت (Disparity Image) من صورتين مجسمتين (Pair of Stereo Images).

في هذه الطريقة نقوم بمطابقة البكسلات في الصورة الأولى مع مقابلاتها في الصورة الثانية، وكنتيجة نقوم بالمحافظة على الاختلافات الحادة في التفاوت (Disparity)، بالإضافة إلى عدد قليل جدا من الأخطاء الناتجة ولكن ذلك في سبيل تسريع عملية حساب الـ Disparity Image.

1-3-5-مطابقة خطوط المسح Matching Scanlines:

يتم ترميز التطابق (Correspondence) كسلسلة من التماثلات بين البكسلات (match sequence)، كل تماثل (match) هو عبارة عن زوج مرتب (x,y) يعبر أن شدة السطوع (Intensity) البكسلين x و y، $I_L(x)I_R(y)$, على الترتيب، هي صورتين لنفس النقطة في المشهد (سوف نعتبر أن x هو بكسل من الصورة اليسرى و y هو بكسل من الصورة اليمنى). في حال الحصول على بكسلات في الصورة الأولى ليس لها بكسلات مقابلة تحقق معها تماثل (match) في الصورة الثانية فإنه يتم تشكيل ما يسمى بالـ Occlusion، وهو عبارة عن مجموعة البكسلات المتتالية في خط المسح المحصورة بين بكسلين لهما تماثلين في الصورة الثانية أو بين بكسل له مطابق وحدود الصورة. كمثال على عملية المطابقة بين خطوط مسح مختلفة(انظر الشكل 4).

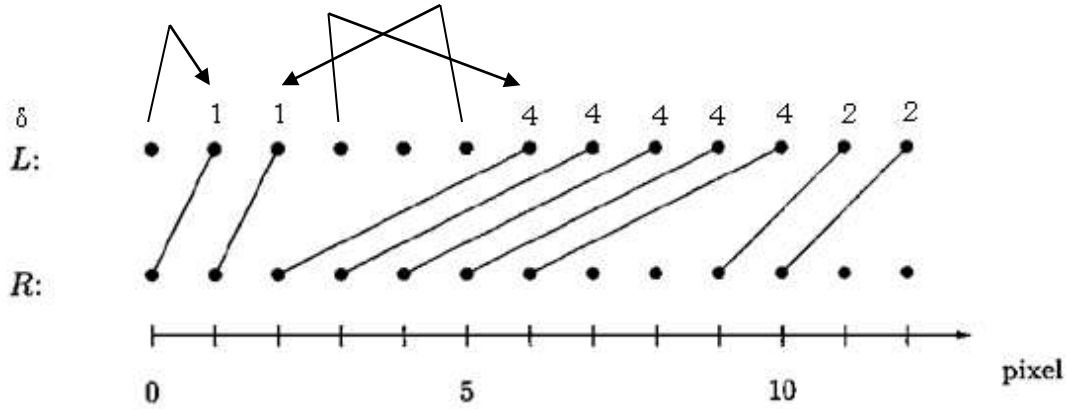


The match sequence

$$M = \langle (1,0), (2,1), (6,2), (7,3), (8,4), (9,5), (10,6), (11,9), (12,10) \rangle.$$

الشكل(4): تحديد التماثلات (Matchs) وتحديد الـ Occlusions

التفاوت للبكسل x (ويرمز له بـ $\delta(x)$) والموجود في الصورة اليسرى والذي يوجد له تماثل (Match) مع البكسل y في الصورة اليمنى يعرف كما يلي: $x - y$ (عملية الطرح تمثل الفرق بين موضع كل من البكسلين)، حيث تمثل x موضع (الموضع: هو ترتيب البكسل يأخذ قيم من 0 إلى n-1 حيث n تمثل عدد البكسلات في خط المسح) البكسل في خط المسح الموجود في الصورة اليسرى، وتمثل y موضع البكسل في خط المسح الموجود في الصورة اليمنى. أما بالنسبة للبكسلات المؤلفة للـ Occlusion (هذه البكسلات هي البكسلات التي لم يتحقق تماثل مع بكسلات في الصورة الأخرى) تأخذ قيمة التفاوت البكسل الأبعد من البكسلين الذين لهما تماثل واللذان يحدان منطقة الـ Occlusion. يوضح الشكل () مثال عن قيم التفاوت للبكسلات في بالنسبة لخط المسح اليساري للصورة.



The match sequence

$$M = \langle (1,0), (2,1), (6,2), (7,3), (8,4), (9,5), (10,6), (11,9), (12,10) \rangle.$$

الشكل (5): توضيح قيم التفاوت Disparity بالنسبة لكل بكسل في خط المسح اليساري

5-3-2- تابع الكلفة Cost Function:

مع كل سلسلة من التماثلات M (Match Sequence) نربط معها كلفة $\gamma(M)$ التي تقيس مدى تعبير السلسلة عن التطابق الحقيقي بين البكسلات (حيث أفضل السلاسل التي تعبر عن التطابق هي السلسلة ذات الكلفة الأقل). تعرف كلفة سلسلة التطابق بقيمة جزائية (Penalty) ثابتة لكل Occlusion، وقيمة موجبة لكل تطابق، ومجموع قيم عدم التشابه (Dissimilarity) بين البكسلات المتطابقة كما هو موضح في العلاقة التالية:

$$\gamma(M) = N_{occ}K_{occ} - N_mK_r + \sum_{i=1}^{N_m} d(x_i, y_i)$$

يتضح لنا من المعادلة السابقة المعلومات التالية:

1- N_{occ} يمثل عدد الـ Occlusion.
2- N_m يمثل عدد التماثلات (Matches).
3- كلما زاد عدد الـ Occlusion، فإن كلفة التطابق تزداد والعكس بالعكس كما هو موضح في الحد الأول.
4- كلما زاد عدد التماثلات (Matches)، فإن كلفة التطابق تنقص والعكس بالعكس كما هو موضح في الحد الثاني.
5- يتم إضافة قيم عدم التشابه (Dissimilarity) لكل تماثل إلى كلفة التطابق، حيث كلما زادت قيمة عدم التشابه بين البكسلين المؤلفين للتماثل تزداد كلفة التطابق والعكس بالعكس كما هو موضح في الحد الثالث.

K_{occ} وK_r هي عبارة عن ثوابت. بشكل حدسي يتم اختيار K_{occ} تفسر على أنها الدليل الضروري من أجل تحديد تغير في التفاوت، في حين K_r تفسر على أنها أكبر فرق متوقع Expected Dissimilarity بين البكسلات المشكلة للتماثل،حيث يتم تقدير قيم هذه الثوابت تجريبيا بالاعتماد على البيئة التي يتم العمل بها.

نعرف عدم التشابه Dissimilarity بين بكسلين كما يلي:

$$d(x_i, y_i) = |I_L(x_i) - I_R(y_i)|$$

5-3-3- الشروط على عملية المطابقة بين البكسلات Hard Constraints:

بالإضافة إلى استخدام تابع الكلفة Cost Function من أجل جودة سلسلة التطابق سوف نقوم بإضافة بعض الشروط، تفيد هذه الشروط في:

1- تسهيل الشروط عملية البحث المتناظر في فضاء سلاسل التطابق الممكنة.

2- عدم السماح بوجود أنواع محددة من سلاسل التماثلات (Match Sequences) غير المفيدة.

الشروط هي:

C1: $0 \leq x_i - y_i \leq \Delta, i = 1, \cdots, N_m$

C2: $y_1 = 0$

C3: $x_{N_m} = n - 1$

C4: $x_i < x_j \ \ and \ \ y_i < y_j \ where \ 1 \leq i < j \leq N_m$

C5: $x_{i+1} = x_i + 1 \ or \ y_{i+1} = y_i + 1 \ where \ \ i = 1, \cdots, N_m - 1$

C6: $If \ \langle x_i, \cdots, x_j \rangle \ is \ an \ occlusion \ in \ the \ left \ scanline,$

$then \ x_j \ lies \ to \ the \ left \ of \ intensity \ variation, 1 \leq i \leq j < N_m$

C7: $If \ \langle y_i, \cdots, y_j \rangle \ is \ an \ occlusion \ in \ the \ right \ scanline,$

$then \ y_i \ lies \ to \ the \ right \ of \ intensity \ variation, 1 < i \leq j \leq N_m$

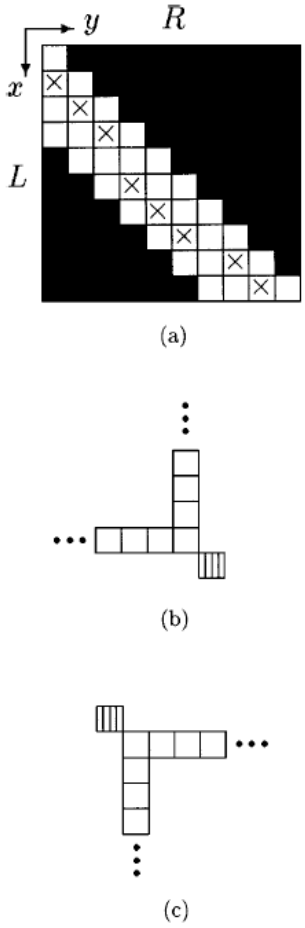
فيما يلي سوف نقوم بشرح هذه الشروط:

1- يفيد الشرط الأول في تصغير فضاء البحث وذلك بتعريف Δ التي تعبر عن القيمة العظمى التي يمكن أن يأخذها التفاوت (Disparity). يفرض هذا الشرط أن تكون قيمة التفاوت غير سالبة.
2- يجبر الشرط الثاني أن يكون البكسل الواقع في أقصى يسار خط المسح الأيمن أن يكون له تماثل Match مع أحد البكسلات في خط المسح الأيسر دون تحديد موضع هذا البكسل.
3- يجبر الشرط الثالث أن يكون البكسل الواقع في أقصى يمين خط المسح الأيسر أن يكون له تماثل Match مع أحد البكسلات في خط المسح الأيمن دون تحديد موضع هذا البكسل.
4- يقوم الشرط الرابع بتحديد خواص الوحدانية والترتيب بين التماثلات، حيث يتم الاستفادة من هذا الشرط من أجل تبسيط البرمجة الديناميكية.
5- يحول هذا الشرط دون وجود Occlusions متزامنة في (أي في نفس المواضع) في كل من خطي المسح.
6- الشرط السادس والسابع يفيدان في معالجة المناطق غير المنقشة من المشهد.

5-3-4-خوارزمية المطابقة The Matching Algorithm:

بالاعتماد على بنية تابع الكلفة الذي قمنا بشرحه سابقا، يمكننا الاعتماد على تقنية البرمجة الديناميكية من أجل إيجاد سلسلة التماثلات الأمثلية.

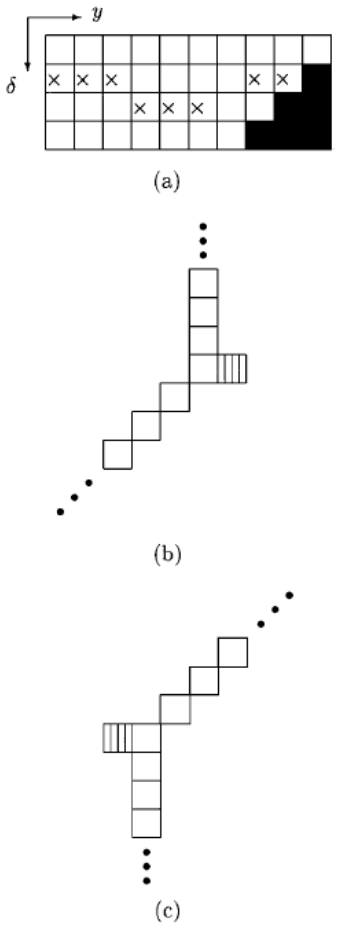
يوضح الشكل(6) شبكة البحث بين خطي مسح يتألف كل منهما مع عشرة بكسلات مع استخدام التفاوت الأعظمي ليكون $\Delta = 3$، أي يأخذ التفاوت بين البكسلات في الصورة قيمة من أربعة قيم 0,1,2,3، من أجل الحصول على سلسلة تطابق.



الشكل(6): فضاء البحث من أجل إيجاد سلسة تماثلات

تمثل الخلية (x,y) تماثل ممكن بين البكسل x في الصورة اليسرى والبكسل y في الصورة اليمنى. نظرا للشرط الأول الذي يحدد قيمة التفاوت الأعظمي $\Delta = 3$، فإن هنالك العديد من الاحتمالات غير المسموحة في عملية التماثل. أي سطر أو عمود لا يحوي على علامة X فإن ذلك يدل على أن البكسل المقابل لهذا السطر أو العمود occluded pixel. لأي تطابق (x_i,y_i) فإن التطابق الذي يمكن أن يسبقه مباشرة (x_{i-1},y_{i-1}) يمكن أن يكون أحد الخلايا كما هو موضح في الجزء (b) من الشكل (6) وذلك بالاعتماد على الشرطين الرابع والخامس. وبالمثل فالتطابق الذي يمكن أن يليه مباشرة (x_{i+1},y_{i+1}) يمكن أن يكون أحد الخلايا كما هو موضح في الجزء (c)من الشكل (6).

يبين الشكل (7) النسخة المضغوطة من فضـــاء البحث. يصبح الإحـــداثي العمودي بدلا من x هو $y = x - y$ والذي يدل على التفاوت.



الشكل(7): فضاء البحث المضغوط للفضاء الموجود في الشكل (6)

لكل خلية من خلايا الشبكة المضغوطة Shifted Grid نقوم بتخزين المعلومات التالية: $\varphi[\delta, y]$ والذي يمثل الكلفة لأفضل سلسلة تماثلات وصلنا إليها في هذه المرحلة والتي تنتهي عند التماثل$(y + \delta, y)$ كما نقوم بتخزين التطابق السابق المباشر لهذه الخلية$\pi[\delta, y]$.

بسبب وجود العلاقة الثنوية بين التماثلات (Matches) السابقة لتماثل معين وبين التماثلات التالية له فإن يوجد خوارزميتان ثنويتان من أجل البحث في فضاء التماثلات.

الخوارزمية الأولى هي حدسية ومباشرة أما الخوارزمية الثانية تعطينا إطار يمكن العمل عليه وتفيدنا في تسريع عملية البحث عن طريق تشذيب الخلايا غير المفيدة.

إذا جعلنا $\gamma_0(x,y)$ تعبر عن كلفة أفضل سلسلة تماثلات التي تنتهي عند التماثل (x,y)، فإن الخوارزميتان تقومان بالبحث من اليمين إلى اليسار ومن الأعلى إلى الأسفل ضمن الشبكة المضغوطة φ، وتقومان بحساب تكلفة الطريق الأفضل لكل خلية كما هو موضح فيما يلي:

$$\varphi[\delta,y]=\gamma_0(y+\delta,y)$$
$$=d(y+\delta,y)-\kappa_r+\min\begin{Bmatrix}\varphi[\delta,y-1],\\ \varphi[\delta-1,y-1]+\kappa_{occ},\dots,\\ \varphi[0,y-1]+\kappa_{occ},\\ \varphi[\delta+1,y-2]+\kappa_{occ},\dots,\\ \varphi[\Delta,y+\delta-\Delta-1]+\kappa_{occ}\end{Bmatrix}$$

حيث يتم حساب الـ Minimum على كلف الطرق الأمثلية للتماثلات الســـابقة للتماثل $(y+\delta,y)$، هذه الطرق هي:

1- الطريق الذي لا يحوي على أي occlusion.
2- الطريق الذي يحوي على left occlusion.
3- الطريق الذي يحوي على right occlusion.

بعد تعبئة مصفوفة الكلف φ، يتم اختيار الطريق ذو الكلفة الأصغر، الذي يحقق الشرط الثالث ضمن الشروط التي ذكرناها سابقا.

الخوارزمية الأولى والتي يطلق عليها The Backward-Looking Algorithm تمر على كل خلية $[\delta,y]$ من خلايا الشبكة المضغوطة والتي تقوم بحساب الكلفة المحددة في المعادلة السابقة، عند الوصول إلى خلية معينة فإنه يتم معاينة جميع التطابقات السابقة لهذه الخلية $[\delta_p,y_p]$ من أجل تحديد أي خلية تقع الطريق الأمثلي، من هنا جاء اسم هذه الخوارزمية نظرا لأنها تبحث عن الخلايا التي تسبق الخلية الحالية.

الخوارزمية الثانية والتي يطلق عليها The Forward-Looking Algorithm تقوم بتقسيم عملية minimization الموجودة في المعادلة السابقة حيث يتم حساب $\varphi[\delta,y]$ بشكل تسلسلي (Sequentially) كما يلي:

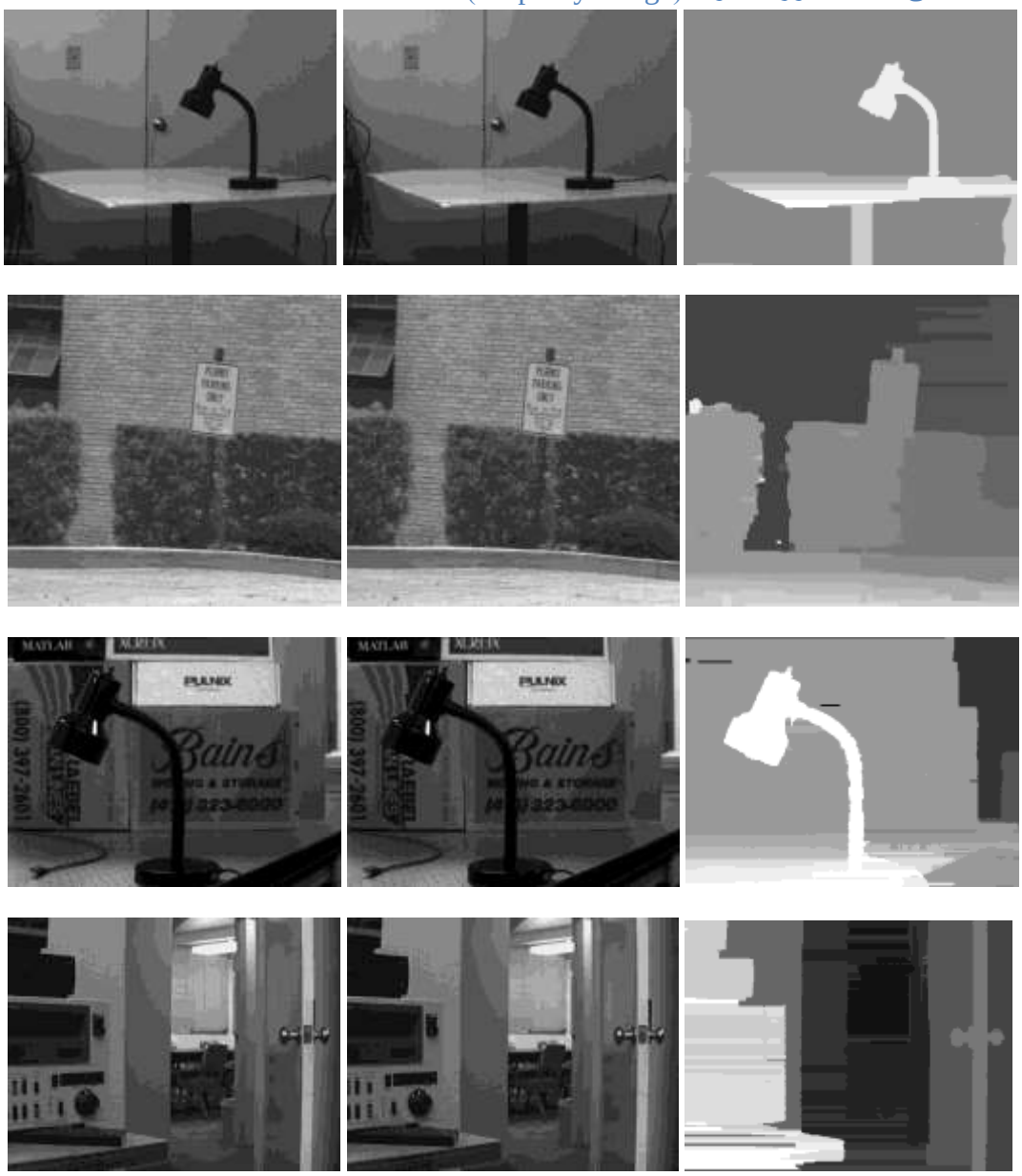
1- يتم حساب العنصر الأصغري بين المعاملين الأولين.
2- يتم المقارنة بين القيمة الناتجة والقيمة والثالثة ومن ثم مع القيمة الرابعة وهكذا.

تقوم بالتكرار على الخلايا $[\delta_p,y_p]$ من الشبكة محددة فيما إذا كانت تقع على الطريق الأفضل لأحد التماثلات $[\delta,y]$ التي تليها. الطريق إلى الخلية لا يتم تحديثه وإنما يتم تحديث الطريق إلى التماثل التالي الممكن. اكتسبت هذه الخوارزمية اسمها نظرا لأنها تبحث عن الخلايا التي تلي الخلية الحالية.

الخوارزميتان السابقتان تقومان بنفس العمليات الحسابية من رتبة $O(n^2)$ حيث يمثل Δ التفاوت الأعظمي Maximum Disparity. نظرا لأن الخوارزمية الأولى هي حدسية أكثر من الخوارزمية الثانية، فإنه لا يتم ملاحظة فائدة الخوارزمية الثانية. عندما نصل إلى خلية معينة

أثناء العمل باستخدام الخوارزمية الثانية فإن الطريق الأفضل الواصل إليها يكون قد تم حسابه وبالتالي يمكننا قبل أن نتوسع في حسابات الخلية أن نقرر فيما إذا كانت تقع على الطريق الأمثلي أو لا، بالقيام بهذه المعاينة نقول قد شذبنا بعض الخلايا ذات الكلفة العالية التي قد تؤدي إلى حلول غير مفيدة. تقوم الخوارزميتان إضافة إلى ما سبق بنقل المعلومات بين خطوط المسح وذلك لأنه في حال اعتمادنا على المطابقة بين كل خطين متقابلين في الصورة نكون قد ألغينا جزء كبير من المعلومات التي تنقلها الصورة حيث أن كل خطوط الصورة تكون مترابطة مع بعضها البعض.

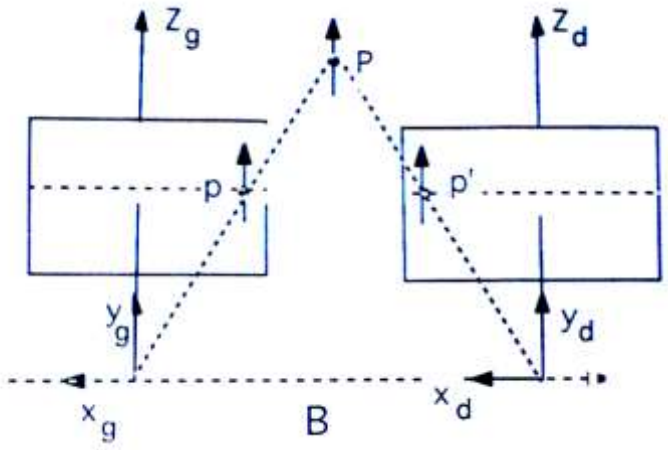
5-3-5- أمثلة على حساب صورة التفاوت (Disparity Image):



الشكل(8): حساب Disparity Image لعدة مشاهد

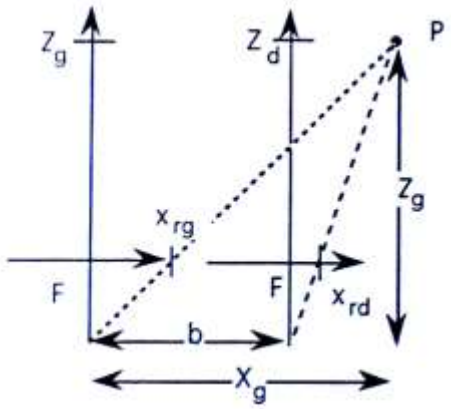
5-4- حساب صورة العمق:

كما رأينا من المرحلة السابقة فإنه بواسطة خوارزمية الـ Pixel-to-Pixel نكون قد حسبنا صورة التفاوت Disparity Image و هي عبارة عن صورة رمادية تتعلق درجة السطوع فيها بمدى قرب العائق (Obstacle) من الروبوت. أما مهمة هذه الخوارزمية فهي استنتاج العمق باستخدام درجة السطوع هذه. فليكن لدينا الشكل (9) والذي يمثل توضع آلتي التصوير:



الشكل(9): تمثيل نموذج الرؤية المجسمة رياضيا

فسوف يتم حساب العمق (z value) لنقطة ما بالاعتماد على تشابه المثلثات باستخدام الشكل (10):



الشكل(10): كيفية حساب العمق هندسيا

حيث يمكن كتابة:

$$\frac{X_g}{Z_g} = \frac{x_{rg}}{F} \text{ و } \frac{X_d}{Z_d} = \frac{x_{rd}}{F}$$

بالطرح بين المعادلتين نجد:

$$\frac{X_g}{Z_g} - \frac{X_d}{Z_d} = \frac{x_{rg}}{F} - \frac{x_{rd}}{F}$$

و بما أن:

$$Z_g = Z_d = Z \text{ و } X_g - X_d = B$$

فنجد:

$$\frac{B}{Z} = \frac{x_{rg} - x_{rd}}{F}$$

$$\Rightarrow Z = \frac{B \times F}{\Delta x}$$

نلاحظ أن معادلة العمق مستقلة عن قيم الـ x_{ig}, x_{id} وتتعلق فقط بقيمة الـ Δx والتي هي درجة السطوع، أي قيمة التفاوت disparity عند هذه النقطة.

5-5- بناء الخريطة Map Building:

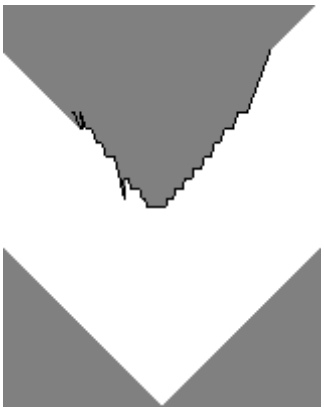
سوف نشرح الآن بشكل مبسط عملية استنتاج الخريطة الصغيرة من صورة العمق و كيفية دمج هذه الخريطة الصغيرة مع الخريطة الكلية للبيئة**[1]**.

5-5-1- بناء الخريطة الصغيرة Minimap Building:

الخريطة الصغيرة هي خريطة تعبر عن توضع العوائق في مجال الرؤية الحالي للروبوت. كل بكسل فيها له ثلاث حالات فقط (هذا ينطبق على الخريطة الكلية للبيئة أيضاً) إما أبيض هذا يعني أن البكسل فارغ ولا يحوي على أي عائق أو أسود و هذا يعني أنا هذا البكسل يحوي على عائق أو رمادي هذا يعني أن حالة البكسل غير معلومة.

بدايةً يتم تحويل مصفوفة التفاوت إلى صورة أحادية البعد (1D) بأخذ أكبر قيمة تفاوت Disparity في كل عامود من أعمدة هذه الصورة. يتم إنشاء خريطة جزئية عرضها و طولها يتعلقان بأكبر قيمة عرض (X) و طول (Z) يمكن أن تأخذها قيم هذه المصفوفة الأحادية و ذلك بعد حذف قيم التفاوت التي لا تتجاوز عتبة معينة.

بعد ذلك و بحسب زاوية الرؤية التي تتمتع بها آلة التصوير يتم رسم العوائق بأماكنها الصحيحة على الخريطة الصغيرة و بتحويل جميع البكسلات التي تقع خلف هذه العوائق إلى رمادي لأنها غير مرئية. بذلك نحصل على خريطة جزئية، انظر الشكل (11):



الشكل (11): مثال على خريطة صغيرة

5-5-2- دمج الخريطة الصغيرة مع الخريطة الكلية للبيئة:

أما من اجل إضافة الخريطة الصغيرة لمشهد ما إلى الخريطة الكلية للبيئة، فالإضافة ليست مباشرة. بدايةً يتم تهيئة جميع بكسلات الخريطة الكلية بقيمة 50%، أي أن هذا البكسل ليس عائق ولا فارغ (رمادي)، ومن ثم تدار الخريطة الصغيرة المـــراد إضافتها بحســـــب توجه (Orientation) الروبوت ثم يتم حســـــاب الوضع النسبي (Relative Position) لكل بكسل من بكسلات هذه الخريطة الصغيرة بالنسبة لموقع الروبوت الافتراضي (وهي النقطة الواقعة في أسفل و منتصف الصورة) ومن ثم تتم المحاكمة التالية:

- إذا كان البكسل رمادي فلن يتم القيام بأي تعديل.
- وإلا فسوف يتم حساب بعد هذا البكسل عن موضع الروبوت.
  - إذا كان البكسل أبيض فسوف تتم إضافة قيمة ثابتة k إلى البكسل المقابل في الخريطة الكلية ويتم تثقيل (ضرب) هذا الثابت بقيمة تتعلق عكساً ببعد البكسل عن موضع الروبوت.
  - إلا فسوف يتم طرح نفس القيمة المثقلة السابقة (حالة البكسل الأسود).
- إذا أصبحت قيمة البكسل من الخريطة الكاملة أصغر من قيمة معينة 20% مثلاً فسوف يعتبر هذا البكسل على أنه عائق ويتم تلوينه باللون أسود، و بنفس الطريقة إذا كانت قيمة البكسل أكبر من قيمة معينة 80% مثلاً فسوف يعتبر هذا البكسل على أنه فارغ ويتم تلوينه باللون الأبيض.

6-5- بناء خريطة البيئة المحيطة من أجل عملية التجوالRepresentation for Navigation:

بناء الخريطة (Map Building) هي عملية تهدف إلى تجميع المعلومات عن البيئة المحيطة من الحساسات المرتبطة بالروبوت (في مشروعنا هذا الحساسات هي الكاميرات) في موقعه الحالي ومن ثم معالجة هذه المعلومات من أجل تحديث الخريطة الحالية**[4]**، الخطوات الرئيسية هي:

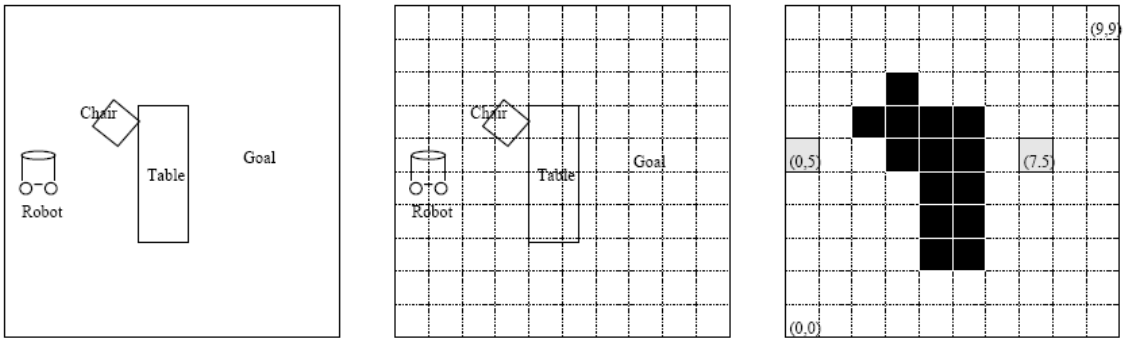
1- الإدراك (Perception): في هذه العملية يتم الحصول على المعلومات من الكاميرات.
2- المعالجة (Processing): في هذا العملية يتم معالجة المعلومات التي حصلنا عليها من أجل بناء تصور لجزء البيئة المحيط بالموقع الحالي للروبوت.
3- الدمج (Fusion): في هذا المرحلة يتم دمج المعلومات التي حصلنا عليها في المرحلة الثانية وتحديث خريطة البيئة المحيطة.

العديد من الطرق تم اقتراحها وتنفيذها من أجل عملية التجوال، الاختـــــلاف الرئيسي بين هذه الطرق هو كيفية تمثيل العالم (البيئة المحيطة). التمثيل "Representation" هو كيفية تخزين المعطيات في ذاكرة الحاسوب وكيفية ربط هذا التخزين مع الأغراض المـــــوجودة على أرض الواقع، سوف نهتم بأحد الطـــــرق التي تعتمد بشكل أساسي على التمثيل الشبكي (Grid-Based Representation)**[3]** .

يتألف العالم الخارجي من مناطق مفتوحة (Open Areas) حيث يستطيع الروبوت أن يتنقل بشكل حر، كما يتألف من مناطق مغلقة (Closed Areas) أو عوائق حيث لا يستطيع الروبوت أن يتنقل. ســـوف نقوم بتمثيـــــل كـــــل من هذه المنـــــــاطق بشبكة احتـــــــلال ثنـــائية البعـد (Two-Dimensional Occupancy Grid)، كل جزء (خلية) من هذه الشبكة يمثل جزءاً صغيراً من العالم الخارجي. يتم تعبئة شبكة الاحتلال بحيث يتم تعليم (Mark) الخلية على أنها فارغة (empty) عندما يكون الجزء المرتبط بها من العالم الخارجي هو تابع لمنطقة مفتوحة، ويتم تعليم (Mark) الخلية على أنها مليئة (full) عندمـــــا يكون الجزء المرتبط بها من العالم الخارجي هو تابع لمنطقة مغلقة، الشكل (12) يوضح مثال عن كيفية تمثيل مشهد معين في شبكة الاحتلال. إن شبكة الاحتلال مناسبة للاستخدام بالنسبة للروبوتات عندما توجد تعديلات في البيئة، بالإضافة إلى سهولة استخدامها في عملية التخطيط. العيب الوحيد في هذا التمثيل هو سعة الذاكرة اللازمة لتخزين هذه الشبكة وخاصة عند الحاجة لتخزين شبكة ذات دقة كبيرة.

عندما تكون الخريطة معروفة فإن شبكة الاحتلال (Occupancy Grid) تقرأ من ملف عند بداية العمل، ولكن كما في مشروعنا يمكن أن تكون هذه الخريطة غير متوفرة وأحيانا يمكن أن تكون خاطئة، لكن لحسن الحظ فإن هذه الشبكة سهلة التعديل في حال وجود اختلاف بين الخريطة وأرض الواقع، فعلى سبيل المثال إذا اكتشف الروبوت وجود عائق جديد في شمال الموقع الحالي، بما أن شمال الموقع الحالي هو الاتجاه Y+، فإنه يتم تعديل الشبكة كما يلي:

```
occupancy [robot_x][robot_y + 1] = full
```



الشكل(12): طريقة تمثيل البيئة باستخدام شبكة الاحتلال

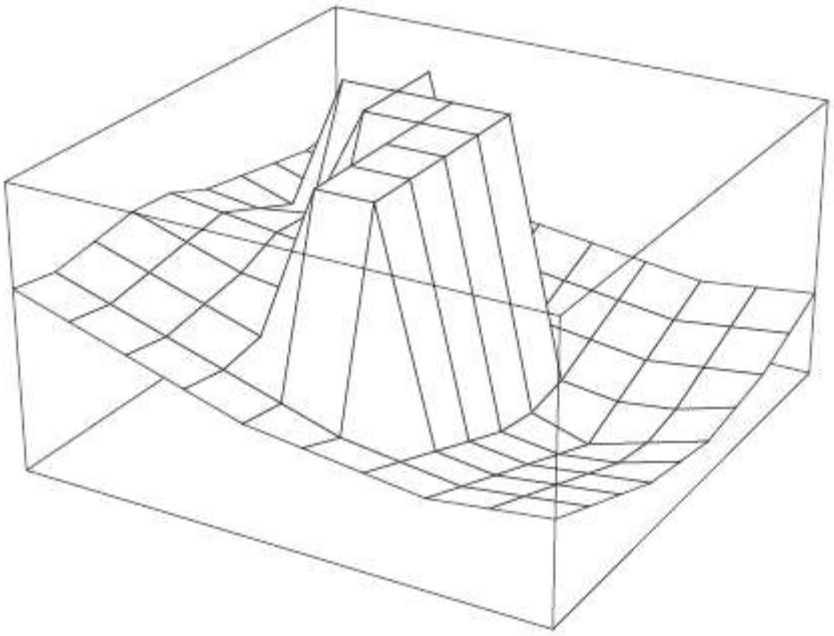
أحد القضايا المتعلقة بشبكة الاحتلال (Occupancy Grid) هي الدقة (Resolution). إن دقة الشبكة تبين كبر المنطقة من أرض الواقع التي تمثلها الخلية في الشبكة، ففي حال كانت الدقة منخفضة فإن ذلك سوف يؤدي إلى طرق سيئة. يعتمد اختيار الدقة على نوع الحساسات ودقتها وعلى سرعة حركة الروبوت وعلى سعة الذاكرة المتوفرة، من المستحسن استخدام دقة عالية في تمثيل الشبكة لأن ذلك سوف يؤدي إلى تمثيل دقيق للعوائق كما سوف ينتج طرق جيدة، ولكن يوجد بعض الأسباب التي تجنب استخدام دقة عالية للشبكة، إذ إنه من غير المنطقي استخدام دقة عالية تفوق دقة حساسات الروبوت، كما أنه في حال كانت سرعة الروبوت تؤدي إلى المرور على عدة خلايا مرة واحدة فإن ذلك يعني أن الدقة عالية وغير ضرورية. يتم عادة تحديد الدقة على أنها المسافة التي سوف يقطعها الروبوت في دورة حسابية واحدة.

7-5- تمثيل الخريطة من أجل تحيد مسار الروبوت Plan Representation:

التجوال (Navigation) هي عملية تهدف إلى توليد حركات الروبوت في البيئة بالاعتماد على المعلومات التي حصلنا عليها في عملية بناء الخريطة (Map Building) وهي تتألف من الخطوات التالية:

1- التخطيط (Planning): في هذه العملية يتم الاستفادة من معلومات بناء الخريطة من أجل تحديد الخطوات الممكنة لحركة الروبوت.
2- الحركة (Motion): في هذه العملية يتم التحرك وذلك استنادا على القرار المتخذ في الخطوة السابقة.

الخطة هي شبكة مثقلة بكلف(Cost Grid)، كل خلية في الشبكة هي تقدير لمسافة أصغر طريق يصل بين هذه الخلية والخلية الهدف، تكون الشبكة المثقلة عادة لها نفس دقة شبكة الاحتلال (Occupancy Grid)، ويتم تمثيل الشبكتين على نفس الشبكة. في ما يلي مثال توضيحي عن الشبكة المثقلة (انظر الشكل (13))، نلاحظ أن كلفة خلية الهدف هي 0.0، وكلفة الخلايا التي تحيط بخلية الهدف تزيد كلما ابتعدنا عن الهدف يمكن تخيل الشبكة ممثلة بشكل ثلاثي الأبعاد بحيث يمثل الارتفاع بعد الخلايا عن الخلية الهدف، كما تمثل العوائق على أنها ارتفاعات عالية جدا، ويمكن تخيل الخطة على أنها دحرجة الكرة على الشكل ثلاثي الأبعاد.



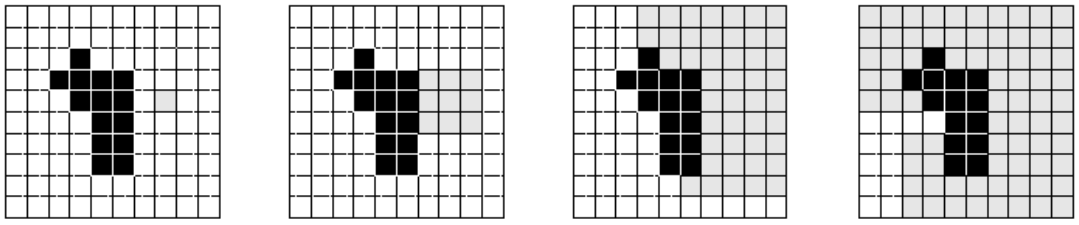
الشكل (13) تمثيل الشبكة بشكل ثلاثي الأبعاد

5-7-1- كيفية حساب الـ Cost Grid:

كما رأينا سابقا فإن الرقم الذي يوجد في الـ Cost Grid يمثل المسافة الأصغر من هذه الخلية إلى الخلية الهدف فيما يلي سوف نشرح كيفية حساب الكلف ضمن الخلايا**[3]**.

في البداية يقوم البرنامج بتهيئة جميع قيم الخلايا بقيمة بدائية، الخلية الهدف وخلايا العوائق لها كلفة ثابتة طيلة فترة الحسابات. إن كلفة الانتقال من الخلية الهدف إلى الخلية الهدف هي 0.0. بما أنه لا يتم الانتقال خلال العوائق فإنها كلفة العوائق تكون قيمتها كبيرة جدا يطلق عليها BIG_COST. الخلايا الخالية يتم تهيئتها في البداية بالقيمة BIG_COST ويتم فيما بعد تغيير قيم الخلايا.

بعد تهيئة قيم الخلايا في الشبكة، يقوم البرنامج بمسح جميع خلايا الشبكة ويقوم بالبحث عن الخلايا التي يستطيع أن يقوم بتغيير كلفها. يتم حساب الكلفة الجديدة عن طريق رؤية الخلايا المجاورة ويقوم بتحديد الكلف عن طريق تقدير الكلفة من الخلية المجاورة إلى الخلية الحالية. كلفة الانتقال إلى الخلية المجاورة أفقيا هي 1. أما كلفة الانتقال إلى الخلية المجاورة قطريا هي $\sqrt{2}$. هاتان المسافتان (القيمتان) هما فقط بالنسبة للانتقال من مركز خلية إلى مركز الخلية المجاورة باعتبار أن ضلع الخلية طوله 1، للحصول المسافة الحقيقة على أرض الواقع فإنه يتم ضرب المسافة بدقة الشبكة يتم تكرار هذه العملية حتى يتم مسح كافة الشبكة.كمثال على هذه الطريقة (انظر الشكل (14)).

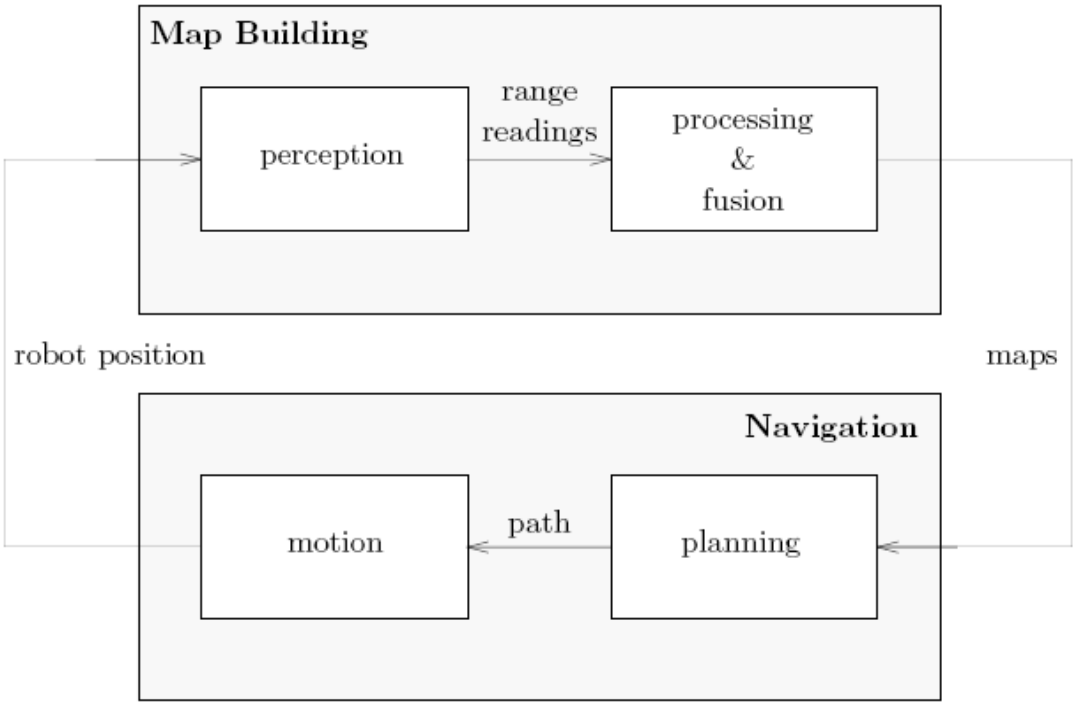


الشكل (14): كيفية حساب كلف الشبكة

تبين الإجرائية التالية كيفية حساب كلف الخلايا في الشبكة:

```
int count = 1;
while (count != 0)
        {
        count = 0;
        for (i = 1; i < GRID_SIZE-1; i++)
                {
                for (j = 1; j < GRID_SIZE-1; j++)
                        {
                        count = count + cell_cost(i,j);
                        }
                }
        }
```

يبين المخطط الموضح في الشكل(15) خطوات بناء الخريطة والتجوال والعلاقة بينهما:



الشكل (15): Map Building & Navigation

6- التصميم الأولي:

في هذا القسم من التقرير سوف نقوم بتحليل النظام ونورد الأفكار والمتطلبات الضرورية لعمله، وسوف نقوم بتحديد الأجزاء العامة التي يتكون منها النظام. يتألف النظام بشكل عام من جزأين أساسيين: جزء عتادي وجزء برمجي وذلك تبعا لطبيعة المشروع التي تتطلب ذلك. النظام هو عبارة عن سيارة (روبوت متنقل) مزودة بآلتي تصوير يتم التحكم بها عن طريق برنامج منصب على حاسوب يتم حمله على السيارة.

6-1- الجزء العتادي Hardware Part:

سوف نتكلم في هذا الجزء عن المكونات العتادية للنظام وخواصها وكيفية التعامل معها. هذه المكونات هي:

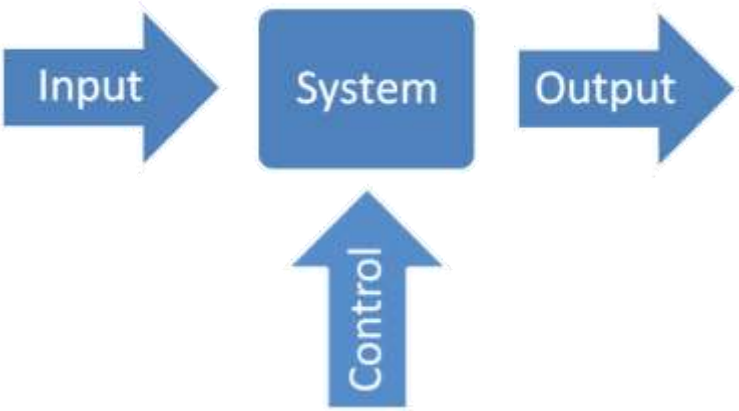
1- نظرا لحاجتنا لروبوت متنقل قمنا ببناء سيارة ولكي تستطيع السيارة العمل دون سائق قمنا بتزويدها ببرنامج يستطيع التحكم بها ويمكّنها من التجول بأمان.
2- حتى يستطيع الروبوت معالجة المعطيات المقدمة له من قبل الحساسات فلا بد من تزويده بوحدة معالجة، لذلك قمنا باستخدام حاسوب تقوم السيارة بحمله كي يقوم بعمليات المعالجة، وإصدار الأوامر للروبوت.
3- يجب تزويد أي روبوت بأداة للإدراك، يتم تزويد الروبوت المتنقل بشكل عام بحساسات تعتمد على تقنية السونار، ولكن في حالتنا فإن الحساسات هي آلتي التصوير التي تمكن الروبوت من التنقّل بشكل آمن ضمن البيئة التي سيعمل بها.
4- نظرا لأنّ الروبوت سيقوم بالتنقّل ضمن البيئة وذلك بعيداً عن الإنسان وسيقوم ببناء خريطة لهذه البيئة فلابد من وجود حاسوب آخر يقوم بإظهار الخريطة المبنية، بالإضافة إلى بعض الأوامر التي يمكن إرسالها من هذا الحاسوب إلى الروبوت عند حدوث بعض الحالات المعينة (توقف لطارئ ما،....الخ).

6-2- الجزء البرمجي Software Part:

سوف نتكلم في هذا الجزء عن المكونات البرمجية للنظام، وسوف نقوم بشرح عام لبيئة النظام البرمجي وكيفية عمله. يتألف هذا النظام من برنامج يقوم بأخذ معطيات من البيئة الخارجية ومعالجة هذا المعطيات ومن ثم يقوم بإصدار الأوامر للروبوت المتنقل. هذه المكونات هي:

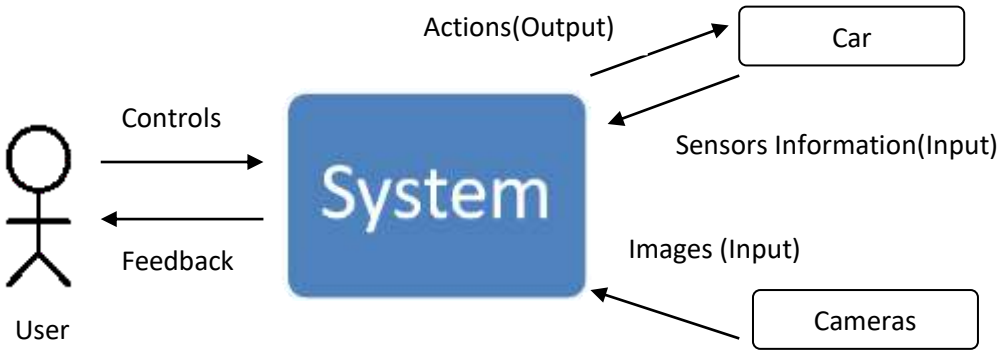
1- كي يستطيع الروبوت العمل بشكل متنقّل لابدّ من تزويده بالقدرة على إدراك البيئة المحيطة واتخاذ القرارات أثناء عمله وذلك عن طريق تزويده ببرنامج يقوم بمعالجة الدخل الآتي من الحسّاسات (آلتي التصوير) ومن ثم يقوم بإصدار الأوامر للروبوت.
2- يحاول الروبوت أثناء تنقّله ضمن البيئة أن يقوم برسم خريطة لهذه البيئة ونظراً لأن البرنامج الذي يقوم ببناء هذه الخريطة هو الجهاز المحمول على السيارة فلابد من وجود برنامج آخر يقوم باستقبال المعلومات من برنامج التحكم ويقوم بإظهار نتائج ومراحل بناء خريطة البيئة.

يمكن بناء المخطط العام للنظام بالاعتماد على الشرح السابق كما هو موضح في الشكل(16):



الشكل(16): المخطط العام للنظام البرمجي

يتم تفصيل المخطط السابق وذلك كما هو موضح في الشكل(17):



الشكل(17): المخطط العام للنظام مع تحديد المؤثرين في النظام (Actors)

اعتماداً على ما قمنا بشرحه في الفقرة السابقة؛ يمكننا التمييز بين جزأين رئيسيين منفصلين هما:

1- الواجهة Interface.
2- النواة Kernel.

الواجهة Interface:

يمثل هذا الجزء من النظام البرمجي واجهة التخاطب مع المستخدم، كما يقوم بإظهار النتائج التي يتم إرسالها من قبل الروبوت كالخريطة المرسومة وغيرها من المعلومات الأخرى. يستطيع المستخدم من خلال الواجهة مراقبة عمل الروبوت وإصدار بعض الأوامر ومنها:

1- اكتشاف Explore:

من خلال هذا الأمر يستطيع المستخدم أن يأمر الروبوت ببدء عملية اكتشاف البيئة المحيطة والقيام ببناء خريطة لهذه البيئة.

2- توقف Stop:

من خلال هذا الأمر يستطيع المستخدم أن يأمر الروبوت بالتوقف وذلك لأسباب معينة.

3- توقف مؤقت Pause:

من خلال هذا الأمر يستطيع المستخدم أن يأمر الروبوت بالتوقف المؤقت عن عمله.

4- إعادة العمل Resume:

من خلال هذا الأمر يستطيع المستخدم أن يأمر الروبوت بالتحرك من جديد ومتابعة عمله.

5- ذهاب إلى Go to:

من خلال هذا الأمر يستطيع المستخدم أن يأمر الروبوت بالذهاب إلى نقطة معينة في الخريطة.

6- تغيير خيارات المراقبة:

من خلال هذا الأمر يستطيع المستخدم أن يتحول من رؤية الخريطة للبيئة إلى نقل حي لما يقوم الروبوت بتصويره.

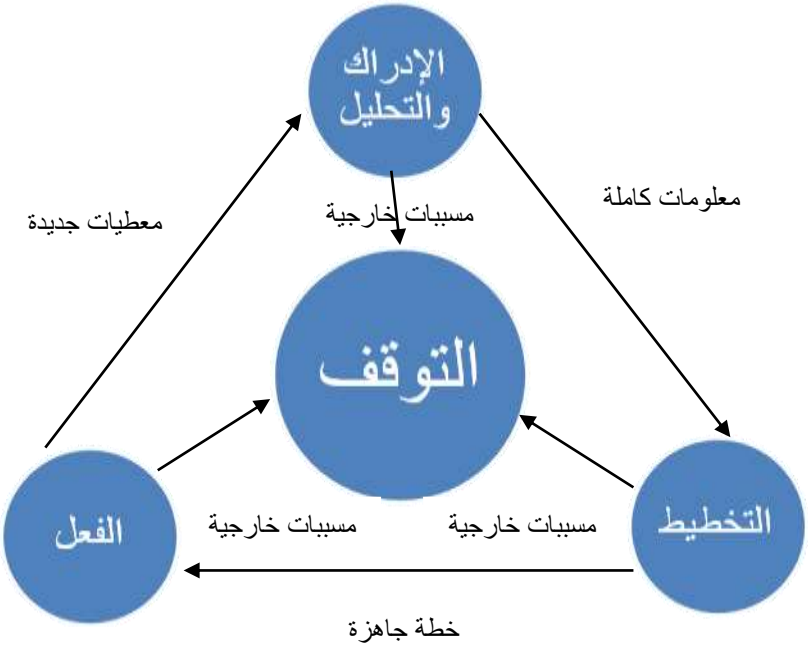
7- تحكم يدوي:

من خلال هذا الأمر يستطيع المستخدم أن يجعل الروبوت يقوم ببعض الحركات (كالتوجه إلى الأمام، أو الرجوع إلى الخلف، أو الانعطاف إلى اليمين أو إلى اليسار...الخ).

النواة Kernel:

يمثل هذا الجزء العقل المفكّر للنظام البرمجي حيث يقوم هذا الجزء بأخذ الأوامر من الواجهة وترجمتها ثم إضافة معلومات الوسط الخارجي عليها (بعد تحليلها) لمعرفة الخطة المناسبة. يقوم هذا القسم بشكل رئيسي بالتحكّم بعمليات تنقّل الروبوت وذلك بالاعتماد على الحسّـــاسات، كما يستقبل بعض الأوامر من المستخدم عن طريق الواجهة.

يبين المخطط الموضح في الشكل (18) حالات النظام التي يمكن أن يأخذها أثناء عمله:



الشكل(18): الحالات التي يمكن أن يأخذها النظام أثناء عمله

نلاحظ من المخطط حالات النظام التالية:

1- **الإدراك والتحليل:**
عندما يكون النظام في هذه الحالة فإنه يقوم بالحصول على الصور من آلتي التصوير ومن ثم يقوم بمعالجتها وعلى أساسها يقوم بإنتاج معلومات من أجل عملية التخطيط.

2- **التخطيط:**
في هذه الحالة يقوم النظام بمعالجة المعطيات الناتجة عن الإدراك والتحليل، ويقوم ببناء الخطة المناسبة لعمل الروبوت.

3- **الفعل:**
في هذا الحالة يقوم النظام بتنفيذ الخطة الناتجة عن حالة التخطيط.

4- **التوقف:**
ينتقل الروبوت إلى هذه الحالة عندما يتوقف بفعل مسببات خارجية أو مسببات داخلية.

7- التصميم التفصيلي:

في هذا القسم من التقرير سوف نقوم بتحديد الخواص التفصيلية لكل من الجزأين العتادي والبرمجي، وذلك لتحقيق البنية العامة التي تم شرحها سابقاً.

7-1- الجزء العتادي Hardware Part:

سوف نتحدّث عن البنية العتادية التفصيلية كما يلي:

7-1-1- بنية الروبوت:

1- آلتي التصوير يجب أن تكونا على قاعدة واحدة والسبب في ذلك: الحاجة إلى أن تتحركا معا وذلك لضمان عدم تغير موقعهما بالنسبة لبعضهما، لأنه لو قمنا بوضع كل آلة على قاعدة مستقلة فإن ذلك سوف يتسبب بحدوث بعض الأخطاء الناتجة عن تغير الوضع النسبي بينهما.

2- يجب أن يتألف الروبوت من عربة واحدة وليس عربتين، وذلك من أجل تسهيل عملية الانعطاف أثناء الحركة وبالتالي لابد من وضع الحاسوب المحمول الذي يحوي النظام البرمجي المتحكم بالروبوت على العربة وعدم جره خلفها.

3- يجب أن يكون حجم الروبوت معتدل لا كبير بحيث يؤدي إلى صعوبة الانتقال في المناطق الضيقة، ولا صغير بحيث يستطيع أن يحمل الحاسوب المحمول.

4- يجب أن يتم توفير المساحة اللازمة من أجل وضع القطع الإضافية كوحدة التحكم بالمحركات وآلات التصوير وبرج الأشعة تحت الحمراء.

5- يجب أن يكون تصميم الروبوت متماسك بحيث لا يتأثر بعلميات الاهتزاز، ويستطيع تحمل أوزان القطع التي يحملها.

7-1-2- ميكانيكية العمل:

- **نظام التعليق:**

لابد لنظام التعليق من أن يكون قادراً على تحقيق الثبات أثناء عمل السيارة، بحيث لا يتأثر عند الانعطاف والتوقف. يجب على نظام التعليق أن يكون قادراً على تحمل عبء الأوزان الناتجة عن الحاسوب المحمول والقطع الإضافية الأخرى. يجب أن تكون المسافة الفاصلة بين العجلات قصيرة نسبياً وذلك لتسهيل عملية الانعطاف وتقليل أخطاء الحركة الناتجة عنها. في حالة الانعطاف فإنه يجب علينا تحريك العجلات الخلفية بسرعتين مختلفتين للتحكم بعملية الانعطاف بدقة.

- **القوة الدافعة:**

يجب أن تكون المحركات قادرة على تحريك الروبوت وسوف يكون الدفع خلفي بالإضافة إلى استخدام نظام تروس من أجل زيادة عزم المحركات لتحريك الروبوت بسهولة أكبر.

2-7- الجزء البرمجي Software Part:

في هذا القسم من التقرير سوف نتحدث عن الخواص التفصيلية للجزء البرمجي من النظام، ولتسهيل عملية التصميم سوف نقوم بتعريف النظام على عدة مستويات، انطلاقاً من المستوي العام الأقل تفصيلا نزولا حتى نصل إلى الأجزاء الصغيرة والوظائف المتعلقة بها. سوف يتم الفصل بين نوعين من المخططات: النوع الأول هو المخططات المتعلقة بحالات النظام (أو العمليات التي يقوم بها النظام) دون ذكر الأجزاء المسؤولة عنه، أما النوع الثاني من المخططات فله علاقة بأجزاء النظام (أي تقسيم النظام إلى وحدات منفصلة) وتحديد مهام كل جزء من هذه الأجزاء.

1-2-7- المستوى الأول من مخطط الحالات(العمليات):

معظم الحالات تتشارك في خطة العمل الموضحة في الشكل(19):

الشكل(19): المستوى الأول من مخطط الحالات

فيما يلي سوف نقوم بشرح المخطط السابق:

المرحلة الأولى: تجميع المعطيات (Data Gathering):

يتم في هذه المرحلة تجميع المعطيات من البيئة الداخلية والبيئة الخارجية للنظام، حيث تمثل هذه المعطيات الأساس في العمليات اللاحقة للنظام. يمكن أن تكون طبيعة المعطيات نصية كالأوامر مثلا، أو تكون عبارة عن صور، أو تكون عبارة عن أرقام تعبر عن قياسات لعوامل معينة ...الخ.

المرحلة الثانية: معالجة المعطيات(Data Processing):

يتم في هذه المرحلة معالجة المعطيات التي تم الحصول عليها في المرحلة السابقة لإنتاج معلومات جديدة يستطيع النظام الاستفادة منها. يمكن أن تكون المعالجة هي عبارة عن تطبيق معادلة ما على معطيات معينة أو معالجة صور واستخراج أغراض من صورة معينة ...الخ. ينتج عن هذا المرحلة معلومات تساعد النظام في اتخاذ قرارات أو القيام بعمليات معينة.

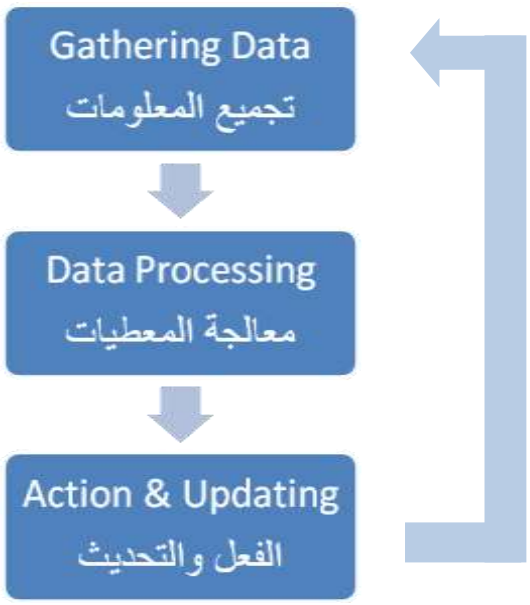
المرحلة الثالثة: التخطيط (Planning):

يقوم النظام في هذه المرحلة باستخدام المعلومات الناتجة عن مرحلة معالجة المعطيات (Data Processing)، من ثم استخدامها لبناء خطة ما من أجل تنفيذ فعل معين أو سلسلة من الأفعال. ينتج عن هذه المرحلة مجموعة من الأوامر التي تمكن النظام من القيام بأفعال معينة كتغير حالته أو البقاء بنفس الحالة.

المرحلة الرابعة: الفعل والتحديث (Action & Updating):

في هذه المرحلة يتم الحصول على الأوامر من مرحلة التخطيط (Planning)، ومن ثم تنفيذ هذه الأوامر. يتم في هذه المرحلة أيضا تحديث المعلومات التي تطرأ عليها بعض التغيرات وذلك نتيجة لتنفيذ أفعال معينة. كما نلاحظ في المخطط السابق أنه بعد الانتهاء من مرحلة الفعل والتحديث فإنه يتم الانتقال مرة أخرى لمرحلة تجميع المعطيات (Data Gathering) نظرا لحالة النظام التي تغيرت نتيجة لتنفيذ الأفعال.

فيما يلي مخطط آخر(انظر الشكل (20)) يشبه المخطط السابق ولكن يوجد اختلاف هو عدم وجود المرحلة الثالثة (مرحلة التخطيط Planning) حيث يمكن اعتبار هذه المخطط يعبر عن قيام المستخدم بالتحكم دون قدرة النظام بعملية التخطيط واتخاذ القرارات (أي يقوم بتنفيذ الأوامر التي يصدرها المستخدم):



الشكل(20): مخطط الاستخدام اليدوي للنظام

سوف نقوم بشرح المخطط الموضح في الشكل(20) بشكل مختصر فقط لتوضيح الفرق عن المخطط السابقة:

المرحلة الأولى: تجميع المعطيات (Data Gathering):

كما هو الحال بالنسبة للمرحلة الأولى في المخطط الموضح في الشكل(19) فإنه يتم تجميع المعطيات من البيئة الداخلية والبيئة الخارجية للنظام. كما ذكرنا سابقا يمكن أن تكون طبيعة المعطيات نصية كالأوامر مثلا، أو تكون عبارة عن صور، أو تكون عبارة عن أرقام تعبر عن قيـــاس لعوامل معينة ...الخ.

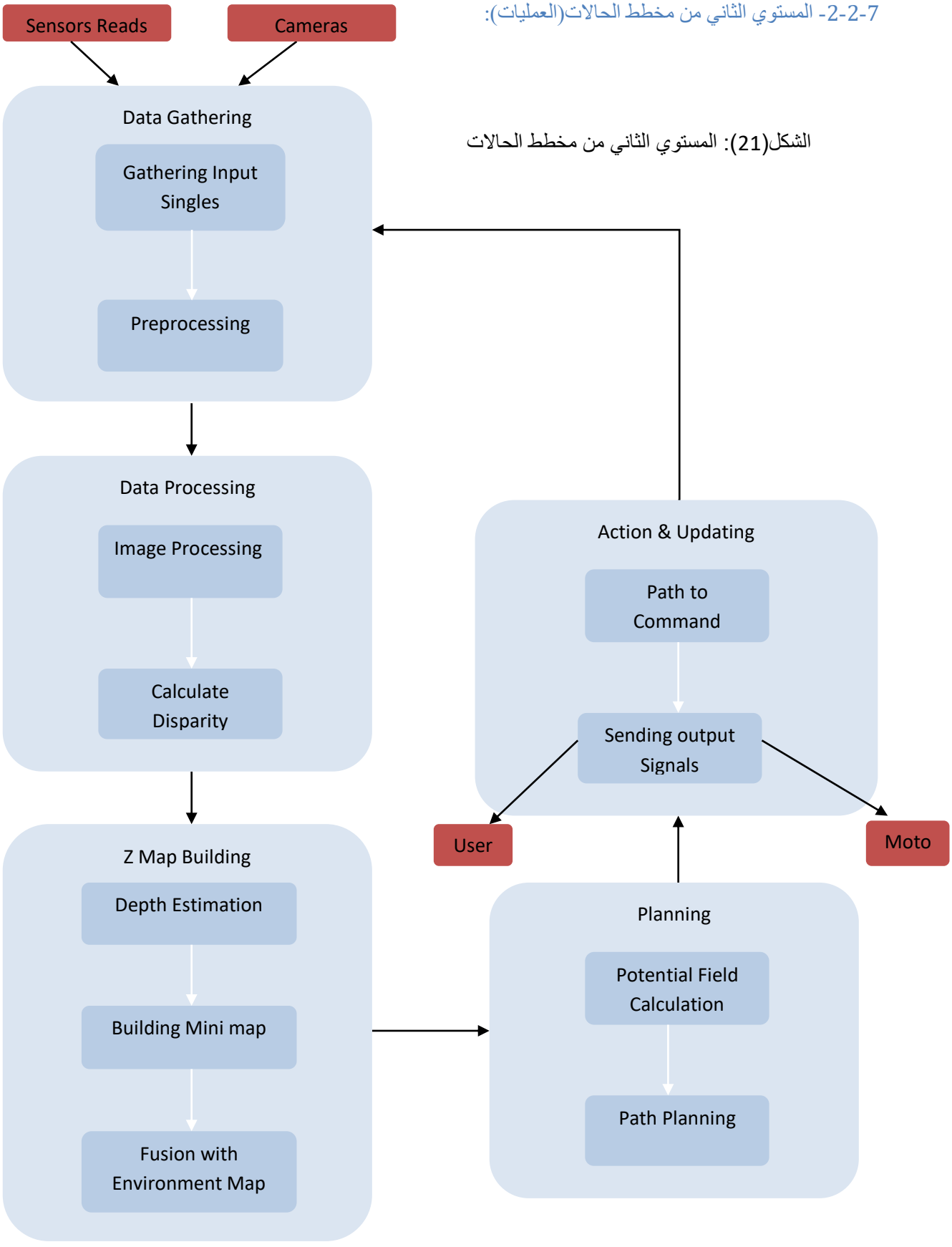
المرحلة الثانية: معالجة المعطيات(Data Processing):

في هذه المرحلة يتم معالجة المعطيات التي تم الحصول عليها في المرحلة السابقة لإنتاج معلومات جديدة ولكن في هذه الحالة يقوم المستخدم وليس النظام بالاستفادة منها. كما ذكرنا سابقا يمكن أن تكون المعالجة هي عبارة عن تطبيق معادلة ما على معطيات معينة أو معالجة صور واستخراج أغراض من صورة معينة ...الخ. ينتج عن هذا المرحلة معلومات تساعد المستخدم في اتخاذ قرارات أو القيام بعمليات معينة.

المرحلة الثالثة: الفعل والتحديث (Action & Updating):

في هذه المرحلة يتم الحصول على الأوامر من مرحلة التخطيط (Planning)، ومن ثم تنفيذ هذه الأوامر. يتم في هذه المرحلة أيضاً تحديث المعلومات التي تطرأ عليها بعض التغيرات وذلك نتيجة لتنفيذ أفعال معينة. كما نلاحظ في المخطط السابق أنه بعد الانتهاء من مرحلة الفعل والتحديث فإنه يتم الانتقال مرة أخرى لمرحلة تجميع المعطيات (Data Gathering) نظرا لحالة النظام التي تغيرت نتيجة لتنفيذ الأفعال.

2-2-7- المستوي الثاني من مخطط الحالات(العمليات):



الشكل(21): المستوي الثاني من مخطط الحالات

نلاحظ من الشكل(21) وجود خمسة مراحل كبيرة تحتوي كل منها على مجموعة من المراحل الصغيرة. فيما يلي سوف نقوم بشرح هذا المخطط:

المرحلة الأولى: تجميع المعطيات Data Gathering:

يتم في هذه المرحلة تجميع المعطيات من البيئة الداخلية والبيئة الخارجية للنظام هذه المعطيات هي أوامر المستخدم (User Commands)، والصور التي يتم الحصول عليها من آلتي التصوير، بالإضافة إلى بعض قيم الحساسات الأخرى (الحساس الدوراني للمحركات). تنقسم هذه المرحلة إلى الجزأين التاليين:

الجزء الأول: تجميع إشارات الدخل Gathering input signals:

في هذا الجزء يتم تجميع إشارات الدخل من الأجزاء المتصلة بالنظام. يتم الحصول على الصور من آلتي التصوير المتصلتان عن طريق واجهة هي Direct Show، كما يتم الحصول على أوامر المستخدم للنظام، بالإضافة إلى قراءات حساسات المحرك ويتم ذلك عن طريقة واجهة برمجية خاصة بالتحكم بالروبوت.

الجزء الثاني: معالجة مسبقة Preprocessing:

حيث يتم في هذا الجزء القيام ببعض العمليات على الدخل الذي حصلنا عليه في الجزء الأول كتطبيق بعض المعادلات وتحويل المعطيات من شكل إلى آخر. تطبّق المعالجة المسبقة على الصور كما تطبّق على قراءات حساسات المحركات.

المرحلة الثانية: معالجة المعطيات Data Processing:

يتم في هذا المرحلة معالجة للمعطيات التي تم الحصول عليها من المرحلة السابقة. تنقسم هذه المرحلة إلى الجزأين التاليين:

الجزء الأول: معالجة الصورة Image Processing:

في هذا الجزء يتم معالجة الصور التي حصلنا عليه من آلتي التصوير حيث في البداية يتم تحويل الصور من صورة ملونة إلى صور رمادية بالإضافة إلى القيام بعمليات تنقية وتشذيب للصور.

الجزء الثاني: حساب التفاوت Calculate Disparity:

في هذا الجزء من يتم الحصول على الصور من الجزء السابق الذي قام بإجراء بعض العمليات عليها ويقوم بحساب صورة التفاوت وهي عبارة عن صورة من تدريجات الرمادي كل درجة من هذه التدريجات تمثل بعد الغرض عن الروبوت. كما رأينا سابقا فإن هذه العملية تمثل قلب عملية الرؤية المجسمة.

المرحلة الثالثة: بناء خريطة العمق Depth Map Building:

يتم في هذا المرحلة الاستفادة من صورة التفاوت التي تم الحصول عليها من المرحلة السابقة وذلك من أجل بناء خريطة للبيئة المحيطة بالروبوت بالاعتماد على أبعاد الأغراض عن الروبوت. تنقسم هذه المرحلة إلى ثلاثة أجزاء هي:

الجزء الأول: تقدير المسافات Depth Estimation:

في هذا الجزء سوف يتم استخدام صورة التفاوت والقيام بعمليات تقدير المسافة عن طريق المعادلات التي تم شرحها في الدراسة النظرية، بحيث ينتج عن هذه العملية بعد الأغراض في المشهد عن الروبوت.

الجزء الثاني: بناء الخريطة الصغيرة Building Mini map:

في هذا الجزء يتم بناء خريطة صغيرة للموقع الحالي (المشهد) للروبوت وذلك بالاعتماد على الأبعاد التي تم الحصول عليها من الجزء السابق.

الجزء الثالث: الدمج مع خريطة البيئة Fusion with Environment Map:

في هذا الجزء يتم أخذ الخريطة الصغيرة التي تم الحصول عليها في الجزء السابق والقيام بعملية دمجها مع خريطة البيئة الكلية وذلك بالاعتماد على المعادلات و العمليات التي تم شرحها في الدراسة النظرية. ينتج عن هذا الجزء خريطة البيئة الكلية المكتشفة إلى الآن من قبل الروبوت.

المرحلة الرابعة: التخطيط Planning:

في هذا المرحلة يتم الاستفادة من الخريطة المبنية في المرحلة الثالثة من أجل القيام بعملية التخطيط للخطوات اللاحقة للروبوت. تمثل هذه المرحلة الأساس في عملية اتخاذ القرار وتحديد الأفعال التي سوف ينفذها الروبوت. إنّ نتيجة هذه المرحلة هي الخطة التي سوف يتبعها الروبوت لاحقا. تنقسم هذه المرحلة إلى جزأين هما:

الجزء الأول: حساب حقل الكمون Potential Field Calculation:

في هذه الجزء يتم حساب حقل الكمون (Cost Grid) للخريطة التي تم بناؤها في المرحلة الثالثة. يتم تحديد نقطة الانطلاق ونقطة الهدف، ويقوم بحساب كمون (كلفة) كل خلية من خلايا الخريطة وذلك بالاعتماد على القوانين والطرق التي تم شرحها في الدراسة النظرية. ينتج عن هذه الجزء الخريطة ولكن بعد حساب كمون كل خلية من خلايا الخريطة.

الجزء الثاني: تخطيط المسار Path Planning:

يتم في هذا الجزء الاعتماد على الخريطة المثقلة الناتجة عن الجزء السابق من أجل بناء المسار الذي يصل بين نقطة البداية ونقطة النهاية. ينتج عن هذا المرحلة الخطة التي سوف يتبعها الروبوت في المرحلة اللاحقة.

المرحلة الخامسة: الفعل Action:

يتم في هذا المرحلة الاعتماد على الخطة التي تم بناؤها في المرحلة الرابعة ويتم معالجة هذه الخطة وتنفيذها. كما يتم في هذا المرحلة تحويل الخطة إلى مجموعة من الأوامر للأجزاء العتادية للنظام. تنقسم هذه المرحلة إلى جزأين:

الجزء الأول: تحويل الخطة إلى مجموعة من الأوامر Path to Commands Conversion:

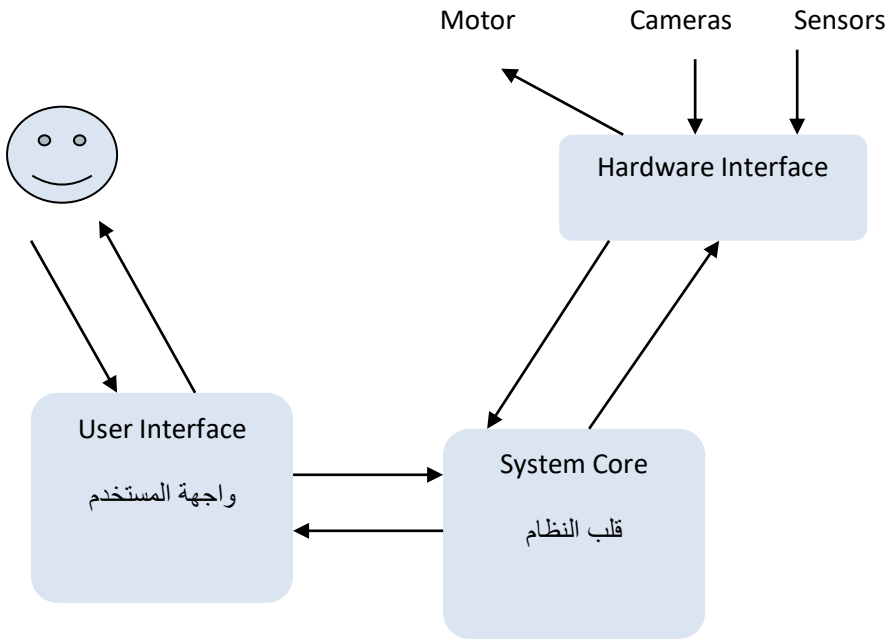
يتم في هذا الجزء استخدام الخطة الناتجة عن المرحلة الرابعة ويقوم بتحويل هذه الخطة إلى مجموعة من الأوامر. يمثل هذا الجزء بدء تنفيذ الخطة.

الجزء الثاني: إرسال إشارات الخرج Sending Output Signals:

يتم في هذا الجزء استخدام الأوامر الناتجة عن الجزء السابق ويتم تحويلها إلى إشارات خرج. هذه الإشارات هي أوامر للمحركات ومعلومات يتم إرسالها إلى واجهة المستخدم لتنبيه المستخدم بما يحدث أثناء عمل النظام. بعد انتهاء من هذه المرحلة يكون النظام قد مر بجميع الخطوات التي تؤلف دورة عمل واحدة. يتم الانتقال من هذه المرحلة إما إلى انتهاء عمل النظام أو إلى دورة عمل جديدة.

3-2-7- بنية النظام System Structure:

سوف نقوم الآن بشرح النظام من وجهة نظر الأجزاء (الوحدات الوظيفية) المؤلفة للنظام، حيث سوف نقوم بشرح كل جزء والمهام المسندة إليه. يبين الشكل (22) المخطط العام للوحدات المؤلفة للنظام.



الشكل(22): الوحدات المؤلفة للنظام

الجزء الأول: واجهة المستخدم User Interface:

هذا الجزء هو المسؤول عن المهام التالية:

1- التخاطب مع المستخدم بحيث يقوم بإظهار النتائج واستقبال الأوامر.
2- التخاطب مع قلب النظام (System Core) حيث يمثل دور الوسيط بينه بين المستخدم.

الجزء الثاني: قلب النظام System Core:

هو المسؤول عن عمليات التخطيط والإدراك ومعالجة الدخل والقيام بتحديد الخرج. يمكننا القول أن قلب النظام مسؤول عن العمليات التالية:

1- إدراك البيئة المحيطة.
2- معالجة المعطيات التي تم الحصول عليها من Hardware Interface.
3- بناء الخطة وتحديد الأوامر.

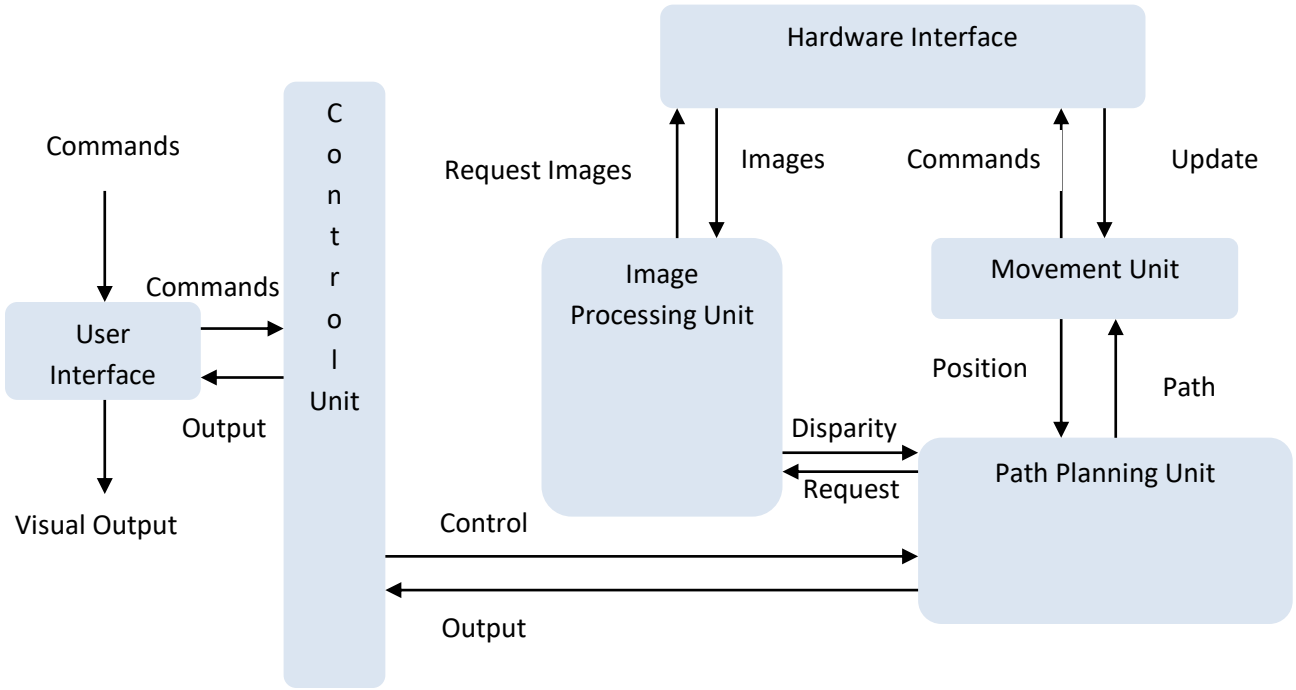
الجزء الثالث: الواجهة العتادية Hardware Interface:

هو يمثل واجهة التخاطب مع آلتي التصوير والمحركات، أي أنه مسؤول عن المهام التالية:

1- تحصيل الصور من آلتي التصوير وإرسالهما إلى قلب النظام لمعالجتهما.
2- إرسال قراءات حساسات المحرك إلى قلب النظام.
3- تحويل الأوامر التي يتم إرسالها من قبل قلب النظام إلى إشارات للتحكم بالمحركات.

7-2-4- المخطط التفصيلي لقلب النظام:

يبين الشكل(23) المخطط التفصيلي لقلب النظام:



الشكل(23): المخطط التفصيلي لقب النظام

7-2-4-1- وحدة التحكم Control Unit:

تمثل هذه الوحدة الأساس في عمل قلب النظام حيث تقوم بإدارة عمليات النظام ونقل الأوامر من واجهة المستخدم إلى الوحدات المسؤولة عن تنفيذها كما أنها تتحكم بالخرج الذي سوف يستقبله المستخدم فيما بعد. يمكن تلخيص مهام هذه الوحدة بما يلي:

1- التحكم بجميع الوحدات الأخرى.
2- التنسيق بين مختلف الوحدات.
3- إدارة عمليات النظام.
4- التعامل مع واجهة المستخدم.

7-2-4-2- وحدة معالجة الصور Image Processing Unit:

تقوم هذه الوحدة بجميع العمليات المتعلقة بمعالجة الصور كما هو واضح من اسمها. بالتالي هذه الوحدة مسؤولة عن المهام التالية:

1- الحصول على الصور من آلتي التصوير عن طريق الواجهة العتادية.
2- معالجة الصور التي تم الحصول عليها.
3- حساب صورة التفاوت.

7-2-4-3- وحدة تخطيط المسار Path Planning Unit:

تقوم هذه الوحدة بجميع العمليات المتعلقة بالخريطة كبنائها وتعديلها، كما تقوم بتخطيط المسار وبنائه. يمكن تلخيص مهام هذه الوحدة بما يلي:

1- حساب صورة العمق Depth Image.
2- بناء الخريطة الصغير Minimap.
3- دمج الخريطة الصغيرة مع خريطة البيئة.
4- حساب حقول الكمون لخلايا الخريطة Potential Field.
5- بناء الخطة اللازمة لعمل الروبوت.

7-2-4-4- وحدة الحركة Movement Unit:

تمثل هذه الوحدة الجزء المسؤول عن عمليات الحركة وتقوم بإرسال الأوامر إلى الواجهة العتادية. هذه الوحدة مسؤولة عن المهام التالية:

1- تقوم بتحويل الخطة إلى مجموعة من الأوامر.
2- إرسال الأوامر إلى الواجهة العتادية Hardware Interface.
3- تقوم بترجمة قراءات الحساسات إلى مواقع على الخريطة.
4- تعمل كوسيط بين الواجهة العتادية ووحدة تخطيط المسار.

8- التحقيق Implementation:

في هذا القسم من التقرير سوف نتحدث عن كيفية تحقيق (Implement) النظام الذي قمنا بشرحه وتصميمه في المراحل السابقة. في البداية سوف نقوم بالتحدث عن تحقيق الجزء العتادي للنظام (Hardware Part) ثم نتحدث عن تحقيق الجزء البرمجي للنظام (Software Part).

8-1- الجزء العتادي Hardware Part:

في ما يلي سوف نتحدث عن كيفية تحقيق الجزء العتادي من النظام. كما ذكرنا سابقا فإن هذا الجزء يتألف من مجموعة من الأجزاء كالمحركات ونظام التعليق بالإضافة إلى آلتي التصوير ووحدة التحكم بالمحركات...الخ.

تم الاعتماد على منصة ليغو ماينستورم (Lego Mindstorm Platform)، من أجل التحقيق العتادي للروبوت المتنقل. تم اختيار منصة ليغو ماينستورم للأسباب التالية:

1- تحتوي على وحدة معالجة (RCX) قابلة للبرمجة ويمكن التخاطب معها بشكل مباشر عن طريق الأشعة تحت الحمراء.
2- جودة التصنيع المتوفرة في الأجزاء العتادية بما يحقق التماسك الجيد والدقة المطلوبة.
3- المرونة في التصميم (إمكانية إنتاج العديد من التصاميم المختلفة باستخدام نفس المصادر).
4- توفر محركات ذات قوة دفع جيدة، بالإضافة إلى وجود مجموعة من المسننات التي تمكننا من بناء نظام تعليق جيد.
5- تتطلب مصادر طاقة بسيطة (بطاريات من نمط AA).

تم تصميم العجلات الخلفية بحيث تحقق تماسكاً عالٍ (تحمّل الأوزان الثقيلة) مع مراعاة التوازن، وتم ذلك عن طريق اعتماد مبدأ نقطتي الارتكاز، بحيث يتوزع الثقل على الطرفين بشكل متساوٍ، وتم تصميم الهيكل الحامل للعجلات بشكل هرمي مما يزيد من تماسك الهيكل بزيادة الوزن الواقع عليه.

تم توزيع قوة الدفع على العجلتين الخلفيتين بحيث يحوي كل طرف محركا منفصلا وذلك لتوليد قوة دفع مثلى تستطيع تحريك وزن السيارة بأكمله، مع اعتماد نظام تروس يزيد من عزم الدوران(يقلل من السرعة). يعتمد هذا النظام على مبدأ نقل الحركة من ترس صغير إلى ترس مجاور كبير، وللاستعاضة عن نقص السرعة تم اعتماد عجلات ذات قطر كبير نسبيا.

بسبب عدم توفر بعض القطع بالكمية الكافية تم استخدام قطع مساعدة (لوح بلاستيكي) وذلك لربط أجزاء السيارة بعضها ببعض بما يحقق تماسكاً جيداً ومساحة جيدة لتوضع الأجهزة على السيارة. يجدر الإشارة إلى أنه تم دراسة توضع العجلات بحيث تتحمل العجلات الخلفية الوزن الأكبر من السيارة وذلك لبنيتها المتماسكة ولتسهيل دوران العجلات الأمامية.

فيما يتعلق بطريقة انعطاف السيارة في الطرق المتعرجة قد تم دراسة تأثير حجم السيارة (الكبير نسبيا) على انعطافها حيث أنه تبين أن عرض السيارة (التباعد بين العجلات) يولد اختلاف في المسافة المقطوعة للعجلات على كل طرف، معنى هذا أنه أثناء انعطاف السيارة ستقطع العجلة على الطرف البعيد من السيارة (بعيد عن مركز الدوران) مسافة أكبر من تلك التي تقطعها العجلة على الطرف القريب مما يعني أنه عليها (العجلة على الطرف البعيد) الدوران بسرعة أكبر من العجلة على الطرف القريب.

يمكن ربط حتى ثلاث محركات بوحدة الـRCX، يستخدم اثنين منها لدفع السيارة والثالث للتوجيه هذا يعني أنه لا يمكننا استخدام محرك رابع لتوجيه البرج الحامل لآلتي التصوير وللتعويض عن هذا النقص تم بناء نظام توجيه مزدوج يربط نظام توجيه البرج بنظام توجيه العجلات، هذا يعني أنه يتم التحكم بكلا من النظامين معاً في آن واحد.

2-8- الجزء البرمجي Software Implementation:

تم اختيار لغة الـ C# تحت بيئة تطوير الـ Net. من أجل تحقيق النظام البرمجي للميزات المتعددة التي تقدمها بيئة التطوير هذه مثل البرمجة غرضية التوجه (Object Oriented Programming)، مرونة الواجهات التي تقدمها بالإضافة إلى قوة تعاملها مع عتاد الحاسوب مثل آلات التصوير، وحدات الدخل والخرج (USB) وتوفير طريقة سهلة لتطبيق الاتصال بين الحواسيب لاسلكيا (Wireless).

تم الاستعانة بالمكاتب التالية في تطوير نظامنا:

1- OpenCV المطورة باستخدام لغة C++ لمعالجات المطورة من قبل شركة Intel وهي عبارة عن مكتبة تحوي الخوارزميات الأساسية لمعالجة الصورة. تم استخدامها لأن تحقيق هذه الخوارزميات ضمن هذه المكتبة هو أمثلي من ناحية وقت التنفيذ والكلفة الحسابية.
2- RCXBrickLibrary وهي عبارة عن مكتبة توفر وسيلة تخاطب فعالة مع وحدة Lego RCX بكامل إمكانيات هذه الوحدة.
3- DirectShow.Net وهي عبارة عن مكتبة تستخدم مكاتب الـ DirectShow والـ DirectX التابعة لنظام التشغيل Windows من أجل التخاطب مع آلتي التصوير في نظامنا.

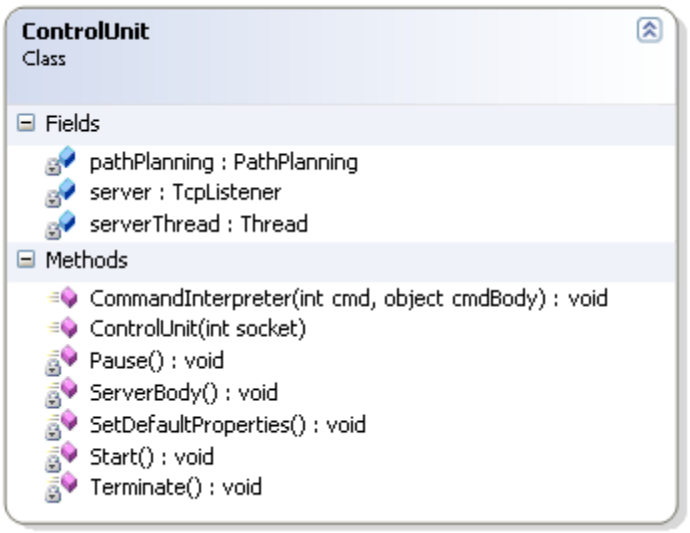
فيما يلي سوف نتحدث عن التحقيق للجزء البرمجي من النظـــــام عن طريق شرح الصفـــوف الأساسية (Main Classes) في البرنامج وطريقة التفاعل والتخاطب بين هذه الصفوف. هذه الصفوف هي:

1- ControlUnit Class
2- PathPlanning Class
3- MovementUnit Class
4- ImageProcessing Class
5- RcxInterface Class
6- CameraInterface Class

يوجد بالإضافة لهذه الصفوف صفوف ثانوية تتعلق بالتعامل مع واجهات التخاطب مع المستخدم User Interface.

سنقوم الآن بشرح مفصل لكل من هذه الصفوف:

1- ControlUnit Class:



هذا الصف هو الصف الأساسي في الجزء البرمجي للنظام الموجود على الحاسب المحمول على الروبوت. يؤمن هذا الصف وسيلة التخاطب اللاسلكية ما بين الحاسوبين وتفسير أوامر المستخدم إلى استدعاء للإجرائيات المسؤولة عن تنفيذ هذه الأوامر.

يحتوي هذا الصف على الحقول (Fields) التالية:

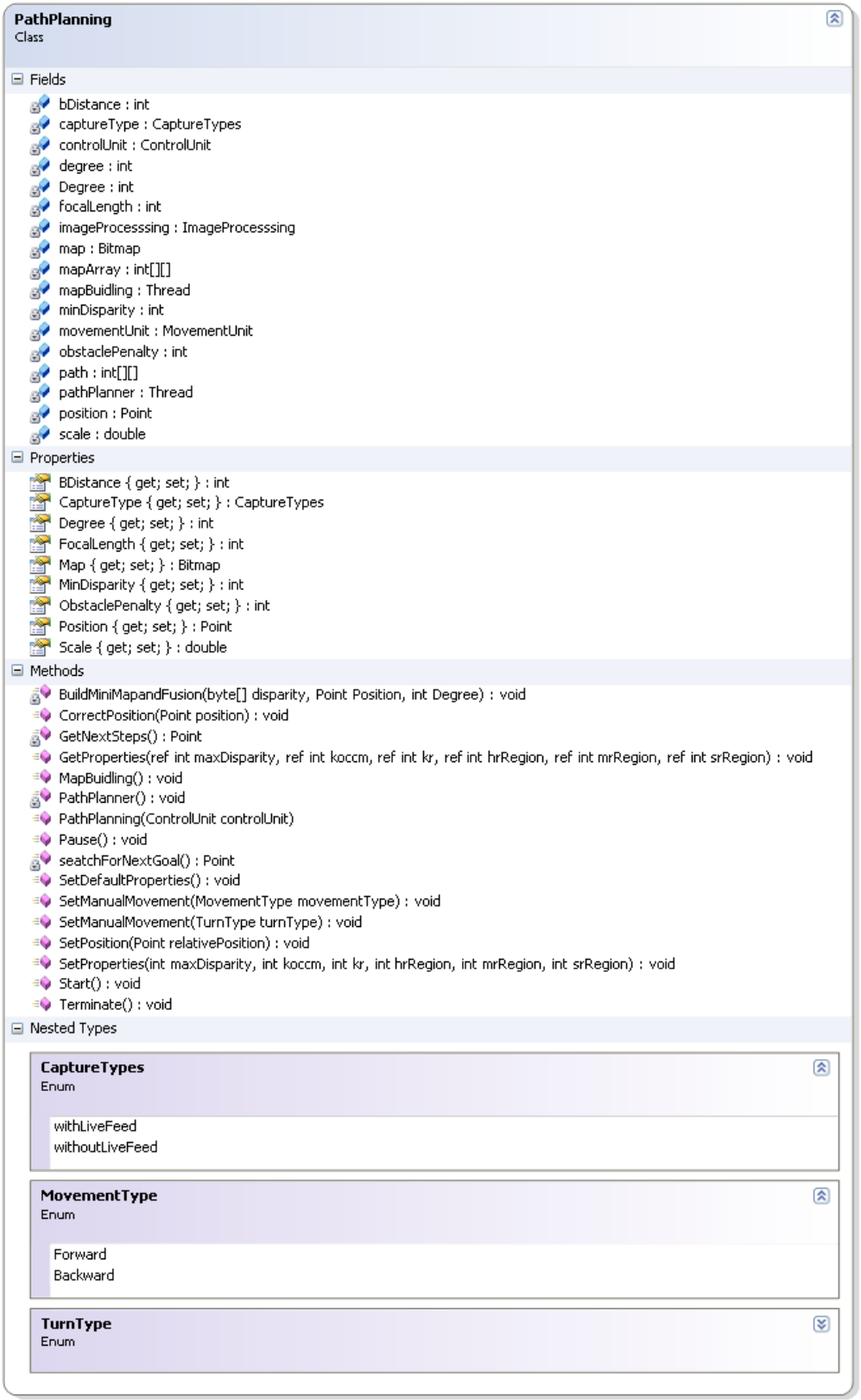
- pathPlanning: هو المسؤول عن جميع العمليات البرمجية من أجل تحقيق خطوات النظام المذكورة في التصميم التفصيلي.
- server: هو المسؤول عن جميع العمليات المتعلقة بالاتصال اللاسلكي باستخدام طريقة الـ Socket.
- serverThread: هو Thread مسؤول عن عملية الانتظار لأي أوامر من المستخدم.

كما يحتوي هذا الصف على الإجرائيات (Methods) التالية:

- الباني Constructor: حيث يأخذ كدخل رقم الـ Socket التي سوف يقوم بعملية التنصت عليها مثلا 5000.
- مجموعة من الإجرائيات البسيطة التي يمكن استدعاؤها دون الحاجة لعمليات تفسير معقدة من قبل المخدم (Server) وهي:
  - Start: هذه الإجرائية مسؤولة عن إعطاء أمر للـ pathPlanning ليبدأ عمليات المعالجة.
  - Pause: هذه الإجرائية مسؤولة عن إعطاء أمر للـ pathPlanning ليتوقف عن عمليات المعالجة في الوقت الحالي مع المحافظة على الوضع الذي وصل إليه.
  - Terminate: هذه الإجرائية مسؤولة عن إعطاء أمر للـ pathPlanning ليتوقف عن عمليات المعالجة نهائياً.
  - SetDefaultProperties: هذه الإجرائية مسؤولة عن إعطاء أمر للـ pathPlanning ليقوم بإسناد الخصائص الافتراضية (Default Properties)لجميع ثوابت الخوارزمية.

- CommandInterpreter: تقوم هذه الإجرائية بتفسير الأوامر المعقدة من قبل المستخدم إلى استدعاءات مناسبة مثال:
  - عندما يطلب المستخدم من النظام القيام بعملية استكشاف Exploration لمدة غير محددة فسوف يتم إرسال أمر الاستكشاف مع متحول قيمته 1- للدلالة على هذا الأمر أوقد يحدد مدة الاستكشاف.
  - عندما يقوم المستخدم بالتحكم اليدوي بالروبوت فسوف يتم إرسال أمر التحكم اليدوي مع الجهة المطلوبة.
  - عندما يقوم المستخدم بتغيير أحد قيم الثوابت فسوف يتم إرسال رمز المتحول المطلوب تغييره مع القيمة الجديدة.
  - عندما يطلب المستخدم من النظام الخريطة الحالية فسوف يتم إرسال رمز للدلالة على هذا الأمر وبالمقابل سوف يرسل المخدم الخريطة المطلوبة.

وهكذا، فدخل هذه الإجرائية نوع الأمر(cmd) وقيمته (cmdBody).


PathPlanning
Class
Fields
bDistance : int
captureType : CaptureTypes
controlUnit : ControlUnit
degree : int
Degree : int
focalLength : int
imageProcesssing : ImageProcesssing
map : Bitmap
mapArray : int[][]
mapBuidling : Thread
minDisparity : int
movementUnit : MovementUnit
obstaclePenalty : int
path : int[][]
pathPlanner : Thread
position : Point
scale : double
Properties
BDistance { get; set; } : int
CaptureType { get; set; } : CaptureTypes
Degree { get; set; } : int
FocalLength { get; set; } : int
Map { get; set; } : Bitmap
MinDisparity { get; set; } : int
ObstaclePenalty { get; set; } : int
Position { get; set; } : Point
Scale { get; set; } : double
Methods
BuildMiniMapandFusion(byte[] disparity, Point Position, int Degree) : void
CorrectPosition(Point position) : void
GetNextSteps() : Point
GetProperties(ref int maxDisparity, ref int koccm, ref int kr, ref int hrRegion, ref int mrRegion, ref int srRegion) : void
MapBuidling() : void
PathPlanner() : void
PathPlanning(ControlUnit controlUnit)
Pause() : void
seatchForNextGoal() : Point
SetDefaultProperties() : void
SetManualMovement(MovementType movementType) : void
SetManualMovement(TurnType turnType) : void
SetPosition(Point relativePosition) : void
SetProperties(int maxDisparity, int koccm, int kr, int hrRegion, int mrRegion, int srRegion) : void
Start() : void
Terminate() : void
Nested Types
CaptureTypes
Enum
withLiveFeed
withoutLiveFeed
MovementType
Enum
Forward
Backward
TurnType
Enum

هذا الصف مسؤول عن التحقيق البرمجي عن جميع المهام الأساسية التي تم ذكرها في التصميم التفصيلي والمتعلقة ببناء الخريطة وحساب أبعاد الأغراض في المشهد وحركة الروبوت.

يحتوي هذا الصف على الخصائص التالي:

- خصائص تتعلق بخوارزمية بناء الخريطة وهي:
  - BDistance: تمثل البعد بين آلتي التصوير.
  - FocalLength: تمثل البعد المحرقي لآلتي التصوير.
  - MinDisparity: تمثل أصغر قيمة للتفاوت يمكن قبولها من أجل حساب العمق.
  - ObstaclePenalty: تمثل أصغر بعد بين نقطتي عائق للوصل بينهما.
  - Scale: تمثل عدد السنتمترات المقابلة لكل نقطة من نقاط الخريطة(مقياس الخريطة).
- Map: هي عبارة عن المتحول الذي يخزن الخريطة كـ Bitmap.
- Position: تمثل الموقع الحالي للروبوت.
- Degree: تمثل التوجه الحالي للروبوت.

كما يحتوي هذا الصف على الحقول التالية:

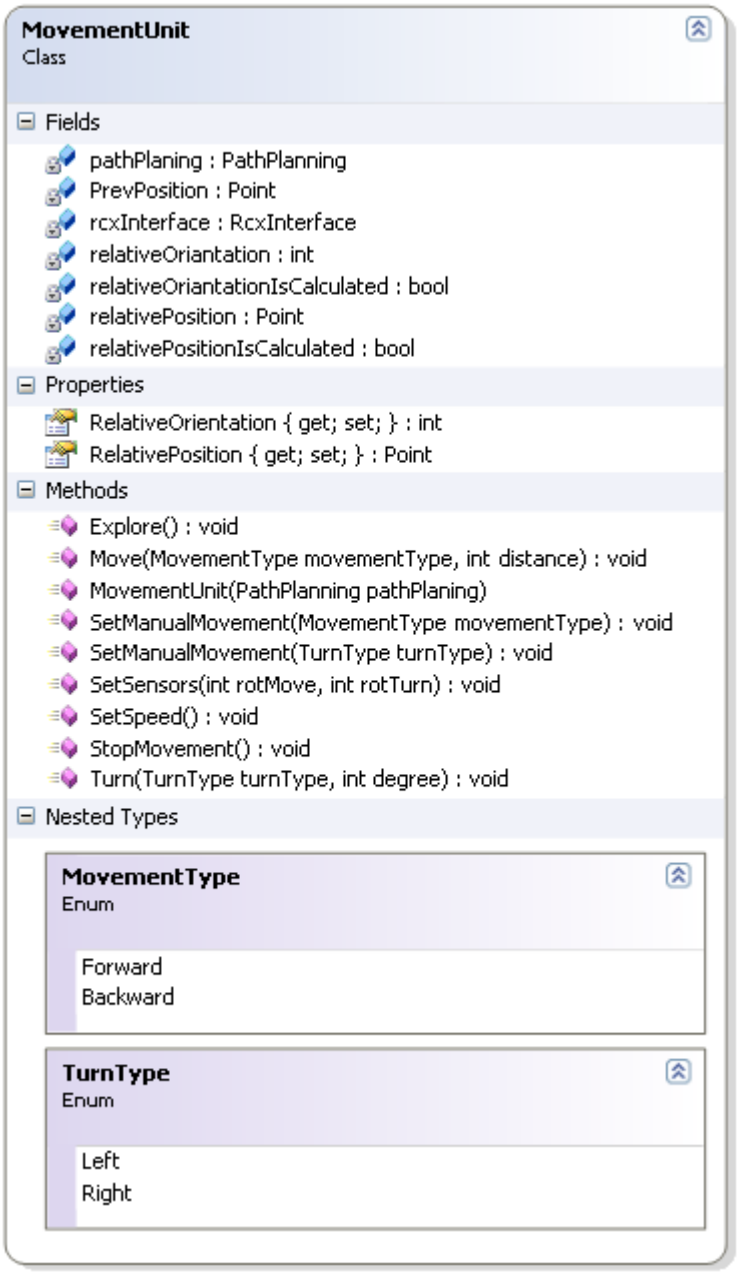
- mapBuilding: هو Thread مسؤول بشكل أساسي عن عملية استدعاء إجرائيات حساب التفاوت من الصف المناسب وحساب الخريطة الجزئية ودمجها مع خريطة البيئة الكلية بشكل دائم.
- pathPlanner: هو Thread مسؤول بشكل أساسي عن عملية التخطيط للحركة ورسم المسار وذلك عند الحاجة إلى ذلك.

كما يحتوي هذا الصف على الإجرائيات الأساسية التالية:

- BuildMiniMapandFusion: وهي إجرائية مسؤولة عن عملية بناء الخريطة، فهي بكل بساطة تطلب صورة التفاوت من صف الـ ImageProcessing (الذي سوف يتم شرحه لاحقا) ثم تقوم بحساب صورة العمق وبناء الخريطة الجزئية ودمجها مع الخريطة الكلية بما يتناسب مع وضع الروبوت وتوجهه.
- إجرائيات مسؤولة عن عملية تخطيط المسار وهي بالترتيب:
  - PathPlanner: تقوم بحساب المسار التخيلي للروبوت.
  - GetNextSteps: تقوم بتحويل مجموعة محددة من الخطوات التخيلية إلى خطوات فيزيائية يتم إرسالها إلى صف الـ MovementUnit (الذي سوف يتم شرحه لاحقا).
- SetManualMovement: وهي إجرائية يتم استدعاؤها عند التحكم اليدوي بالروبوت.

كما يحتوي هذا الصف على بعض القيم المحددة مسبقاً (Enumerators) من أجل رسائل التخاطب مع إجرائياته. على سبيل المثال الـ MovementType وهي إما إلى الأمام (Forward) أو إلى الخلف (Backward).

3- MovementUnit Class:

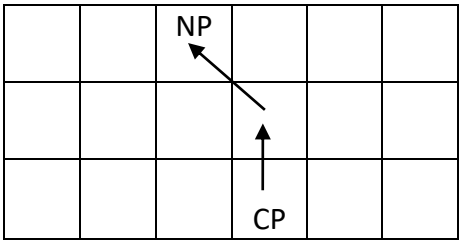


هذا الصف هو عبارة عن صف وسيط ما بين الصفين PathPlanning Class والذي يقوم بحساب المسار وإعطاء أوامر عالية المستوى للتحريك، وبين الصف RcxInterface الذي يتعامل بشكل مباشر مع العتاد (المحركات وحساساتها).

يحتوي هذا الصف على الحقول الخصائص التالية:

- RelativePosition: وهي تعبر عن مدى تغير موقع الروبوت بالنسبة للموقع السابق، أي في حال تحرك الروبوت إلى الأمام فهي سوف تخزن القيمتين x = 0، y=1، وهي تدل على تحرك الروبوت خطوة إلى الأمام.
- RelativeOrientation: وهي تعبر عن الزاوية ما بين وضعه السابق ووضعه الحالي.

مثال:

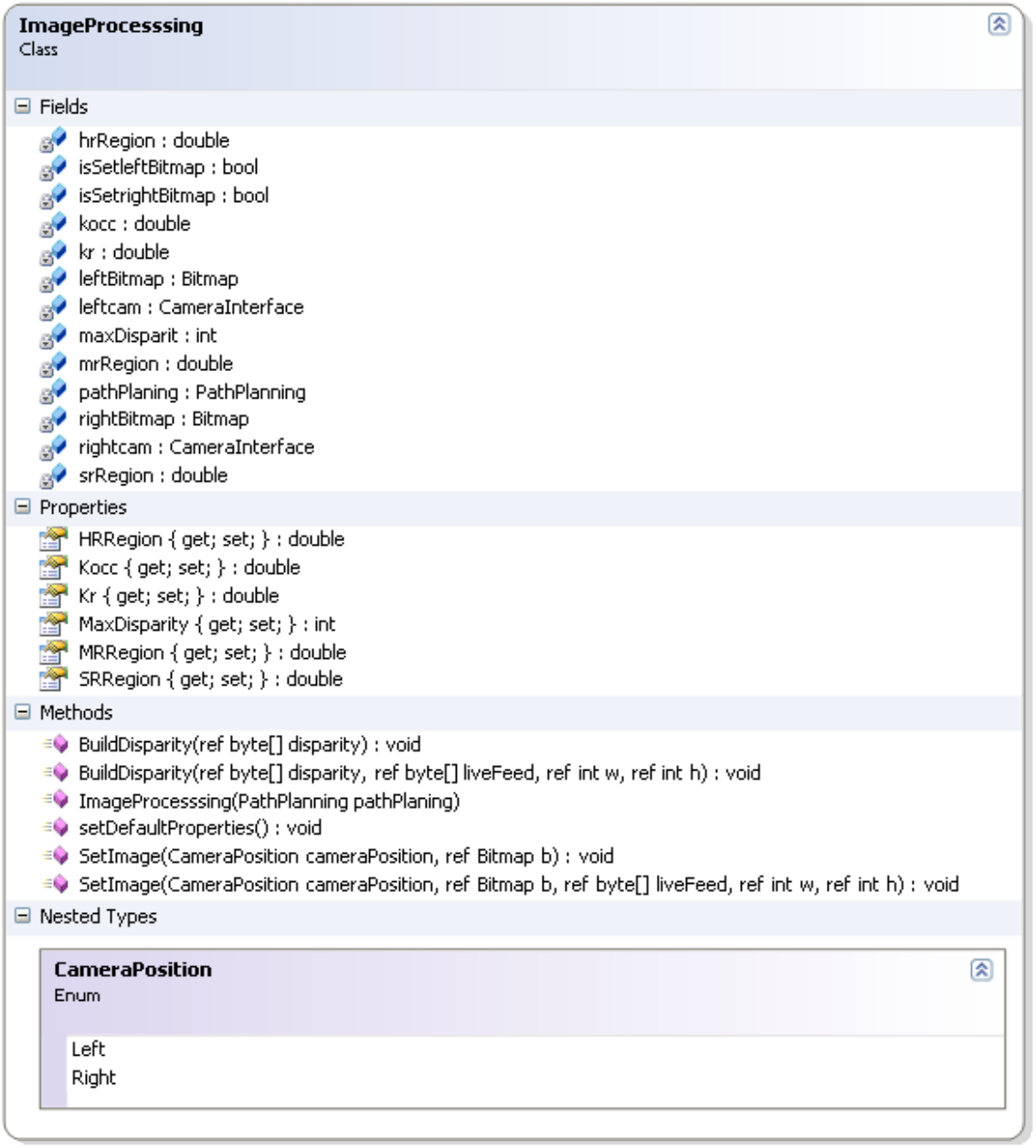


حيث نجد في المثال السابق بأن الـRelativePosition هي x = -1, y = 2 و الـRelative Orientation هي 45 درجة، حيث CP هي الموضع الحالي Current Position و NP هي الموضع الجديد New Position.

كما يحتوي على الإجرائيات التالية:

- SetManualMovement: إجرائية من أجل عملية التحريك اليدوي (Manual Movement).
- Move: وهي إجرائية تقوم بتحريك الروبوت باتجاه معين (إلى الأمام أو الخلف) لمسافة معينة.
- Turn: وهي إجرائية تقوم بتدوير(Rotate) باتجاه معين (إلى اليمين أو اليسار) بدرجة معينة.
- StopMovement: إجرائية من أجل إيقاف الحركة الحالية.
- SetSpeed: تقوم بتحديد سرعة الروبوت.
- Explore: تقوم هذه الإجرائية بالوضع الحالي للروبوت بإيقافه ومن ثم تدوير العجلات إلى أقصى اليمين ومن ثم إلى أقصى اليسار من أجل عملية استكشاف المنطقة الحالية (آلتي التصوير موصولتان على العجلات كما ذكرنا سابقا).

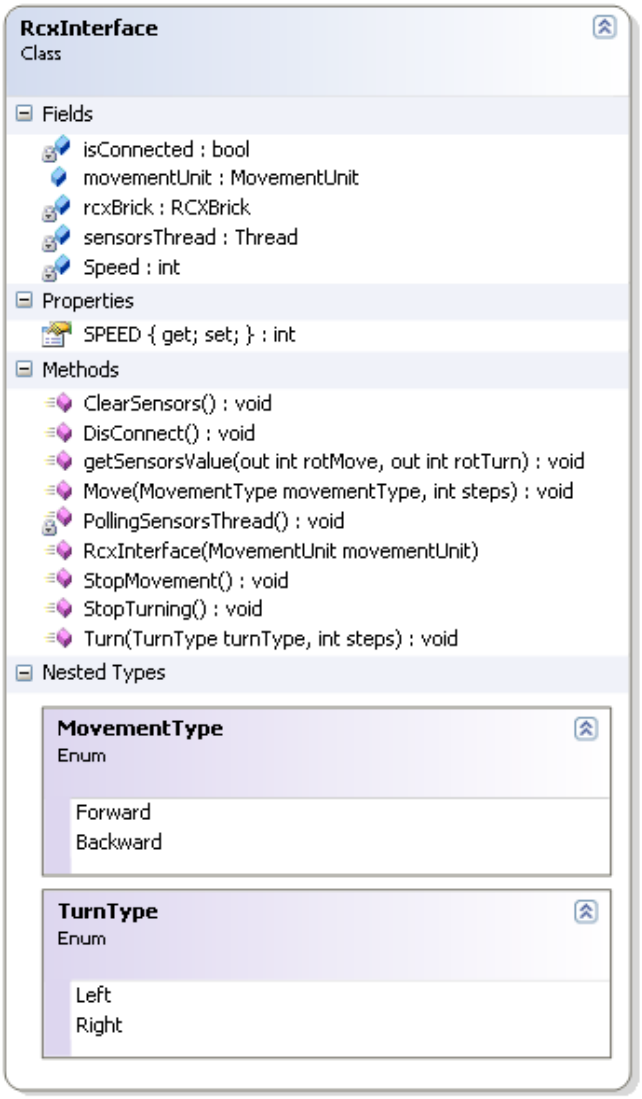
4- ImageProcessing Class:



هذا الصف مسؤول عن عملية حساب صورة التفاوت (Disparity Image).

- يحتوي هذا الصف على مجموعة من الخصائص المتعلقة بالتعامل مع ثوابت خوارزمية حساب صورة التفاوت والتي تم شرحها في الدراسة النظرية، مثلا Max Disparity والتي تمثل الحد الأعظمي للتفاوت بين بكسلين.
- يحتوي هذا الصف على حقلين أساسيين كل منهما يعبر عن آلة تصوير والصورة الناتجة عن هذه الآلة.
- يحتوي على الإجرائية الرئيسية في هذا الصف هي BuildDisparity وهي التي يستدعيها صف الـ PathPlanning.

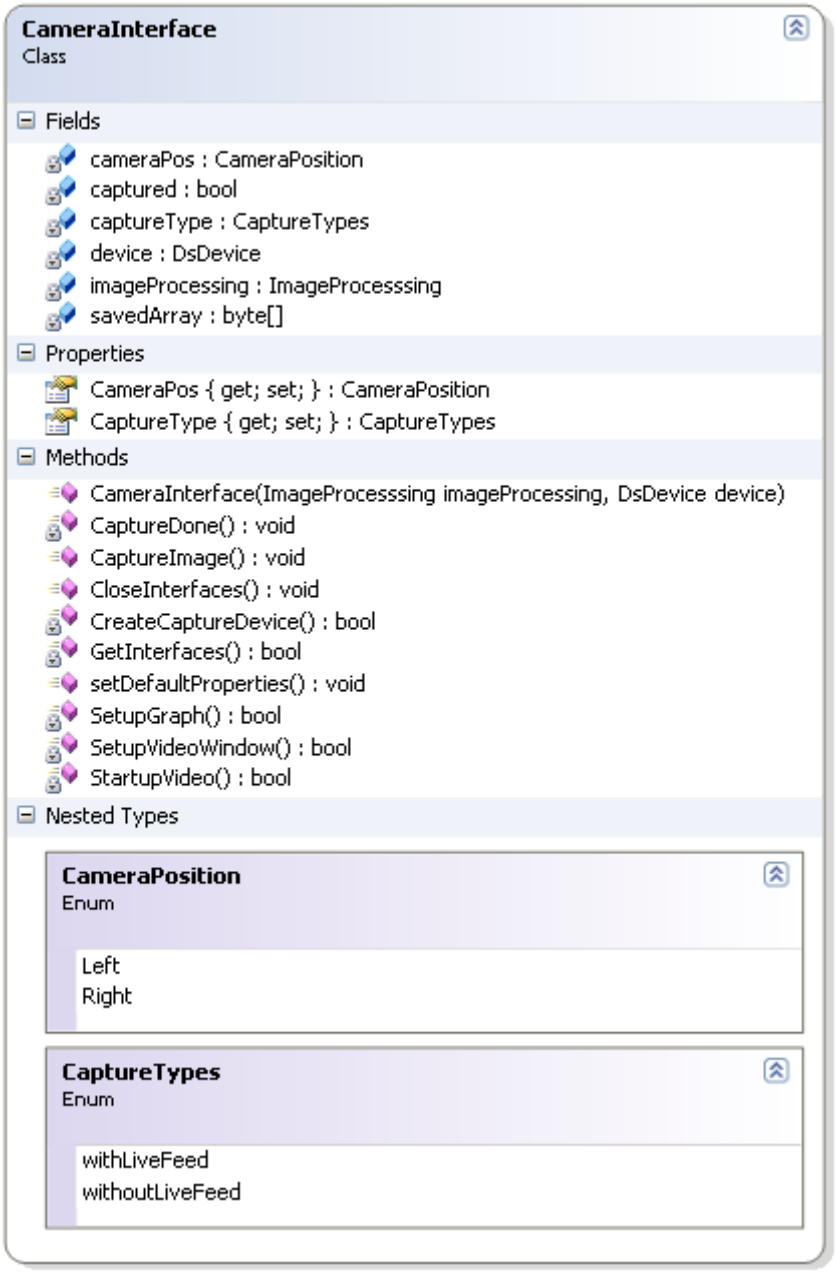
5- RcxInterface Class:



هذا الصف مسؤول عن عمليات التخاطب على مستوى العتاد (Hardware Level).

- يحتوي هذا الصف على خاصة واحدة وهي الـ SPEED والتي تعبر عن سرعة المحركات حيث تأخذ قيم بين 1 إلى 7.
- يحتوي هذا الصف على حقل رئيسي وهو الـ rcxBrick وهو يعبر عن وحدة الـ Lego RCX الفيزيائية بالإضافة إلى sensorThread وهو Thread مسؤول عن عملية استقصاء حساسات المحركات بشكل مستمر ودائم.
- يحتوي هذا الصف على إجرائيات هي: Move، StopMovement، Turn، StopTurning، لها نفس معنى إجرائيات الصف MovementUnit (التي تحمل نفس الاسم) ولكن تتعامل على أساس الخطوات وليس المسافات أي يقوم صف الـ MovementUnit بتحويل المسافة 20 cm على سبيل المثال إلى 4 خطوات محرك فقط (على فرض أن الخطوة هي عبارة عن 22.5 Degree دوران وفي كل دورة يكون الروبوت قد تحرك خمسة سنتمترات).

6- CameraInterface Class:



هذا الصف مسؤول عن عمليات التخاطب على مستوى العتاد مع آلة تصوير.

- يحتوي على خاصة أساسية CameraPos وهي موضع آلة التصوير (Left or Right).
- يحتوي على حقل أساسي هو Device الذي سوف يتخاطب معه.
- يحتوي على إجرائية أساسية CaptureImage وهي التي تقوم بتحصيل الصورة من آلة التصوير، بالإضافة إلى بعض الإجرائيات المسؤولة عن عملية التنصيب (Setup) لآلة التصوير بالطريقة يدركها نظام التشغيل Windows.

9- الاختبارات والنتائج:

منذ بداية العمل على المشروع حتى الوصول إلى التصميم والتحقيق النهائي له كان هناك العديد والعديد من المراحل التي تم المرور بها وصولا إلى المرحلة النهائية. حيت تم القيام بالعديد من التجارب والاختبارات سواء كان ذلك في الجزء العتادي أو الجزء البرمجي للنظام.

كان تصميم السيارة هو الجزء الذي نال القسط الأكبر من عمليات الاختبار والتعديل في الجزء العتادي من النظام، حيث كانت الاختبارات تتركز علي ما يلي:

1- تصميم نظام تعليق السيارة، واختباره من حيث قدرته على تحمل أعباء أوزان الأغراض الأخرى كالحاسوب المحمول ووحدة التحكم بالمحركات بالإضافة إلى القطع الأخرى.
2- تركيز عبء الوزن على المحاور الخلفية للسيارة، من أجل تخفيف الوزن على نظام التوجيه والمرتبط بالمحاور الأمامية وذلك من أجل تسهل عمليات الانعطاف.
3- كيفية توضع آلتي التصوير من أجل التقليل من الأخطاء الناتجة عن الاهتزازات.
4- كيفية تصميم نظام التروس وذلك من أجل توفير العزم اللازم لتحرك الروبوت.
5- كيفية تصميم نظام التوجيه بحيث يمكن الروبوت من التحرك بشكل دقيق قدر الإمكان.

فيما يلي الصور التي توضح المراحل التي مر بها تصميم السيارة:

فيما يتعلق بالقسم البرمجي فكانت الاختبارات تتركز بشكل أساسي على نموذج الرؤية المجسمة Stereo Vision Model، وخاصة في الجزء المتعلق بحساب صورة التفاوت Disparity Image. كما تركزت الاختبارات على كيفية بناء الخريطة والتجوال في البيئة المحيطة بالروبوت.

فيما يلي بعض النتائج المرحلية التي حصلنا عليها من تطبيق الخوارزميات السابقة (خوارزمية حساب Disparity Image، وبناء الخريطة الصغيرة) بالإضافة إلى القيام بعملية بناء لمشهد صغير.

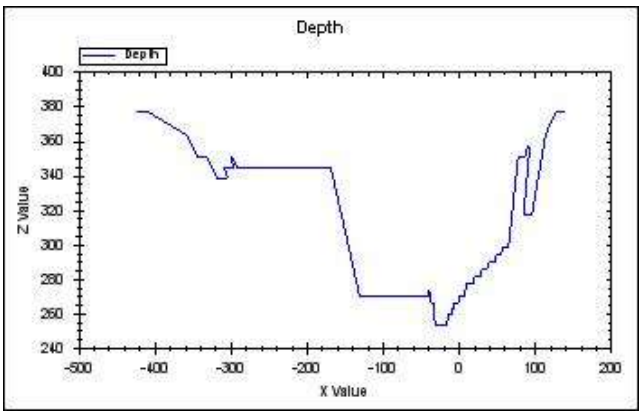
مثال 1:

الصورة اليمينية

الصورة اليسارية

صورة التفاوت

صورة العمق



الخريطة الجزئية

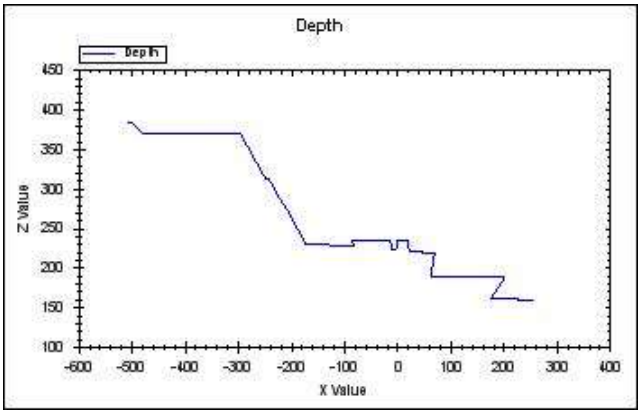
مثال 2:

الصورة اليمينية

الصورة اليسارية

صورة التفاوت

صورة العمق

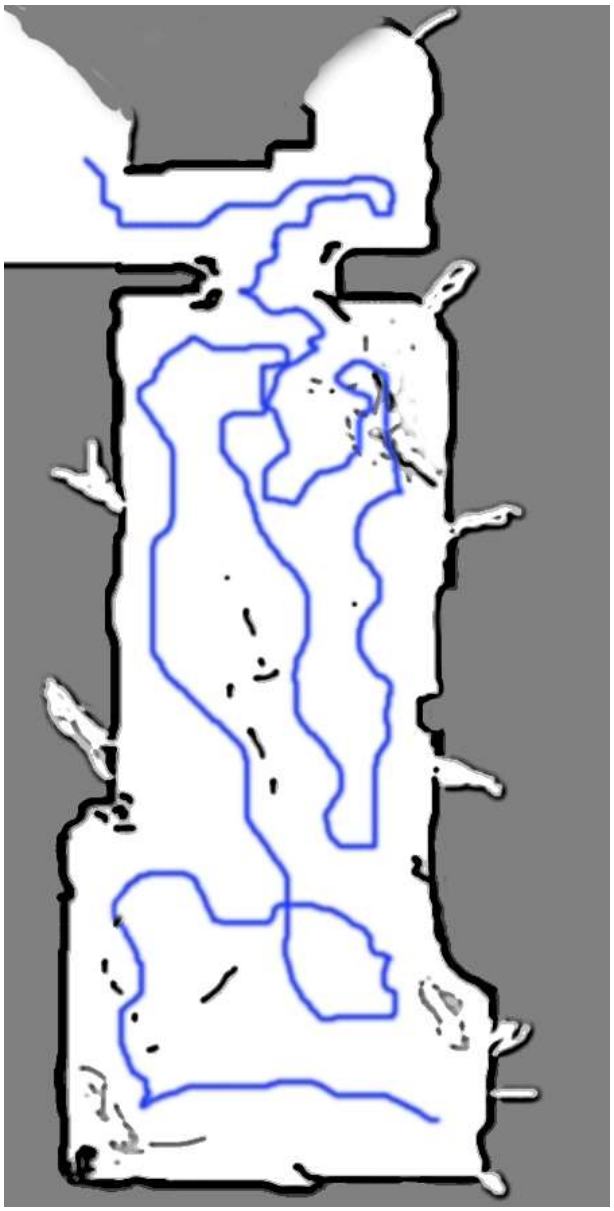


الخريطة الجزئية

مثال 3:

القيام بعملية بناء لخريطة غرفة، فيما يلي بعض الصور التي تم الاعتماد عليه في عملية البناء.

وكانت النتيجة هي الخريطة التالية:



يمكن تلخيص نتائج الاختبارات للنظام ككل بما يلي:

1- تأثر آلتي التصوير بالتغيرات الضوئية في البيئة الخارجية.
2- حدوث بعض الأخطاء في حساب صورة التفاوت Disparity Image وذلك لقلة جودة آلتي التصوير.
3- جودة عمل الروبوت في البيئة الداخلية (Indoor) أحسن من عمله في البيئة الخارجية (Outdoor).
4- تأثر السيارة بوعورة الطريق التي تتحرك عليها مما يؤثر على نظام الرؤية المجسمة وذلك لتأثير الاهتزازات على البرج الذي يحمل آلتي التصوير.
5- بسبب حجم السيارة الكبير نسبيا ونظام التعليق فإن هناك بض الزوايا والمنعطفات الضيقة التي لا تستطيع السيارة العمل بها.
6- وجود أخطاء ناتجة عن قياسات حساسات المحركات.

10- الآفاق المستقبلية:

أثناء إنجاز المشروع وبعد إنهائه، تكونت لدينا آراء حول تحسين النظام بالإضافة لبعض الأفكار لتطويره وإضافة فوائد عملية عليه. تنقسم الآفاق المستقبلية لمشروعنا إلى قسمين هم التحسينات والتطويرات.

10-1- التحسينات:

تهدف التحسينات التي سوف نقدمها إلى زيادة فعالية النظام وزيادة سرعته دون التأثير على الوظائف التي يقوم بها. تتلخص التحسينات فيما يلي:

a. استخدام روبوتات صناعية (هناك العديد منها) وذلك عوضا عن تصميمها وتركيبها من عدة قطع مما يزيد في التماسك والمصداقية والفعالية.
b. استخدام آلات تصوير ذات دقة عالية ومختصة في مجال الرؤية الحاسوبية ليس لجودة الصور التي تعطيها فقط وإنما لقلة تأثرها بالتغيرات الضوئية في البيئة المحيطة.
c. استخدام أكثر من آلتي تصوير (ثلاث آلات تصوير) حيث يتم استخدام آلة التصوير الثالث من أجل المقارنة وتصحيح الأخطاء.
d. استخدام وحدة معالجة أصغر عوضا عن الحاسوب المحمول الذي يشغل مساحة كبيرة ويشكل وزنه عبء على المحركات، واحتوائه عل تجهيزات إضافية غير ضرورية لعمل الروبوت المتنقل.
e. تحسين الخوارزميات المطبقة في النظام واستخدام خوارزميات جديدة ذات كلفة حسابية أقل.

10-2- التطويرات:

تقوم التطويرات على غرار التحسينات بتغيير طرق عمل النظام ولكن بشكل أساسي تهدف إلى زيادة الوظائف التي يقوم بها أو تطوير الوظائف الموجودة فيه. تتلخص التطويرات فيما يلي:

f. استخدام طرق نقل مختلفة كاستخدام تقنية البلوتوث (Bluetooth) عوضا عن الأشعة تحت الحمراء(Infrared).
g. استخدام شبكة عصبونية لها القدرة على تقدر الثوابت المختلفة والتي تعتمد قيما على التجريب وتختلف باختلاف بيئة العمل.
h. استخدام نظام قواعد معرفة (Knowledge Base System)، وذلك من أجل تحسين مقدرة النظام على اتخاذ القرارات أثناء عمله.
i. تطوير طريقة التخاطب مع النظام بحيث يستطيع أن يعالج الأوامر الصوتية من قبل المستخدم عوضا عن الأوامر الاعتيادية.
j. تطوير النظام بحيث يستطيع بناء البيئة بشكل ثلاثي الأبعاد عوضا عن بناء خريطة لهذا البيئة.
k. تطوير النظام بحيث يستطيع المستخدم الحصول على بعد غرض في المشهد وذلك أثناء النقل الحي للبرنامج.
l. تطوير ميكانيكية النظام بحيث يستطيع العمل في بيئة غير مستوية، حيث يوجد اختلاف في ارتفاعات وميل المستويات المؤلفة للبيئة.
m. استخدام حساسات أخرى بالإضافة إلى آلات التصوير، كالحساسات التي تعتمد على تقنية السونار أو الليزر.

11- المراجع:

[1] Don Murray & Jim Little**, *Using real-time stereo vision for mobile robot navigation***, Computer Science Department, University of British Columbia, Vancouver, BC, Canada

[2] Stan Birchfield & Carlo Tomasi**, *Depth Discontinuities by Pixel-to-Pixel Stereo*,** Department of Computer Science, Stanford University, Stanford

[3] Tucker Balch**, *Grid-based Navigation for Mobile Robots***

[4] Giuseppe Oriolo, Giovanni Ulivi, Marilena Vendittelli, ***Real-Time Map Building and Navigation for Autonomous Robots in Unknown Environments***

[5] http://www.wikipedia.com/

[6] http://www.cs.mcgill.ca/

[7] http://vision.stanford.edu/

[8] http://www.vision3d.com/

[9] http://axon.physik.uni-bremen.de/