

**TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN  
KHOA CÔNG NGHỆ THÔNG TIN  
BỘ MÔN THỊ GIÁC MÁY TÍNH VÀ KHOA HỌC RÔ-BỐT**

**TRƯƠNG HOÀNG PHÚC**

**ĐIỀU HƯỚNG ROBOT DỰA VÀO  
CAMERA ĐỒNG BỘ VÀ THIẾT BỊ  
ĐO LƯỜNG QUÁN TÍNH  
CÓ XÉT YẾU TỐ VẬT CẢN**

**KHÓA LUẬN TỐT NGHIỆP CỬ NHÂN CNTT**

**TP. HCM, 2016**

**TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN  
KHOA CÔNG NGHỆ THÔNG TIN  
BỘ MÔN THỊ GIÁC MÁY TÍNH VÀ KHOA HỌC RÔ-BỐT**

**TRƯỜNG HOÀNG PHÚC - 1212296**

**ĐIỀU HƯỚNG ROBOT DỰA VÀO  
CAMERA ĐỒNG BỘ VÀ THIẾT BỊ  
ĐO LƯỜNG QUÁN TÍNH  
CÓ XÉT YẾU TỐ VẬT CẢN**

**KHÓA LUẬN TỐT NGHIỆP CỬ NHÂN CNTT**

**GIÁO VIÊN HƯỚNG DẪN**

**PGS.TS LÝ QUỐC NGỌC**

**KHÓA 2012 - 2016**

## This image shows a full page of primary-ruled paper. It features multiple sets of horizontal dashed lines spaced evenly down the page, providing a guide for handwriting practice. The lines are thin and light gray, set against a plain white background. There are no margins, text, or other markings on the page.

[illegible]

Giáo viên hướng dẫn

[Ký tên và ghi rõ họ tên]

[illegible]

TpHCM, ngày ..... tháng ..... năm .....

Giáo viên phản biện

[Ký tên và ghi rõ họ tên]

# LỜI CẢM ƠN



Xin chân thành cảm ơn các thầy cô thuộc khoa Công nghệ thông tin trường Đại học Khoa học Tự nhiên TP.HCM đã tận tình dạy dỗ, truyền đạt cho tôi nhiều kiến thức quý báu.

Xin chân thành cảm ơn bạn bè đã quan tâm, động viên, và giúp đỡ tôi trong quá trình thực hiện khóa luận.

Xin gửi lời cảm ơn sâu sắc đến Thầy (PGS.TS) Lý Quốc Ngọc vì đã tận tình hướng dẫn, hỗ trợ và góp ý cho em trong quá trình thực hiện khóa luận.

Cuối cùng, tôi xin dành lời cảm ơn sâu sắc nhất xin gửi đến cha mẹ, ông bà vì công ơn sinh thành và giáo dục.

TP. Hồ Chí Minh, tháng 7 năm 2016

*Trương Hoàng Phúc.*

# ĐỀ CƯƠNG CHI TIẾT

<b>Tên Đề Tài:</b> Điều hướng Robot dựa vào Stereo Camera và IMU có xét yếu tố vật cản.
<b>Giáo viên hướng dẫn:</b> TS. Lý Quốc Ngọc.
<b>Thời gian thực hiện:</b> 15/12/2015 đến ngày 15/07/2016.
<b>Sinh viên thực hiện:</b> Trương Hoàng Phúc - 1212296.
<b>Loại đề tài:</b> Nghiên cứu và ứng dụng thị giác máy tính trong lĩnh vực điều hướng robot tự động và tái tạo bản đồ ba chiều.

<p><b>Nội Dung Đề Tài:</b> Khảo sát, nghiên cứu và phân tích một số phương pháp nhằm giải quyết bài toán định vị và xây dựng bản đồ ba chiều. Sau đó, khóa luận đề xuất ra một mô hình để áp dụng cho bài toán trên sử dụng dữ từ camera Stereo và IMU. Hai bài toán chính cần quan tâm đó là ước lượng chuyển động và xây dựng bản đồ ba chiều từ thông tin màu – độ sâu đồng thời phát hiện và né tránh vật cản. Mô hình của khóa luận đề xuất khác biệt so với các cách tiếp cận truyền thống bao gồm bốn bước: định vị vị trí, tái tạo bản đồ lân cận, phát hiện khôi phục lỗi hệ thống và kết hợp với dữ liệu IMU. Khóa luận cũng đã xây dựng được một ứng dụng hoàn chỉnh và đánh giá kết quả trên một bộ dữ liệu đã được công bố.</p>
<p><b>Kế Hoạch Thực Hiện:</b></p> <ul style="list-style-type: none"><li>- 15/12/2015 – 14/01/2016: Khảo sát các nghiên cứu liên quan về định vị và xây dựng bản đồ ba chiều, tập trung vào các nghiên cứu có thể áp dụng ở môi trường ngoài trời.</li></ul>

<ul style="list-style-type: none"> <li>- 14/01/2016 – 15/03/2016: Tìm hiểu các phương pháp ước lượng chuyển động dựa vào thông tin camera Stereo.</li> <li>- 15/03/2016 – 14/04/2016: Cài đặt và kiểm thử thuật toán ước lượng chuyển động dựa vào ảnh từ camera Stereo.</li> <li>- 14/04/2016 – 15/05/2016: Thêm các bước phát hiện và khôi phục lỗi hệ thống.</li> <li>- 15/05/2016 – 30/06/2016: Hoàn chỉnh báo cáo khóa luận, đưa ứng dụng minh họa lên hệ điều hành Windows, đánh giá kết quả trên bộ dữ liệu đã được công bố.</li> </ul>	
<p align="center"><b>Xác nhận của GVHD</b></p>	<p align="center"><b>Ngày.....tháng.....năm.....</b></p> <p align="center"><b>SV Thực hiện</b></p>

# TÓM TẮT

Một hệ thống robot có thể hoạt động hoàn toàn độc lập luôn là mối quan tâm của cộng đồng khoa học nói chung và khoa học máy tính nói riêng. Trong đó, nhiệm vụ then chốt đầu tiên chính là cho robot khả năng tự động điều hướng, muốn được như vậy, robot phải có khả năng biết được vị trí và bản đồ của không gian xung quanh. Đây cũng chính là hướng nghiên cứu nhận được rất nhiều sự quan tâm. Về tổng thể, các mô hình định vị và xây dựng bản đồ có thể được ứng dụng rộng rãi vào thực tế như các nhiệm vụ thám hiểm, trinh sát, làm những công việc nguy hiểm đối với con người hoặc mở rộng ra thành những ứng dụng robot cá nhân như hỗ trợ tìm đường đi, quay phim, chụp ảnh, ...

Nhận thấy được xu thế đó, khóa luận sẽ tập trung nghiên cứu một hệ thống định vị và xây dựng bản đồ ba chiều dựa trên dữ liệu đến từ camera Stereo và IMU. Trên cơ sở ước lượng chuyển động bằng thông tin thị giác cùng với lý thuyết kết hợp nhiều loại cảm biến khác nhau, khóa luận đã áp dụng thành công trong việc xây dựng một mô hình có khả năng ước lượng vị trí robot đồng thời xây dựng bản đồ ba chiều. Đây có thể xem là một phương pháp mới so với các cách tiếp cận truyền thống hiện nay.

Kết quả sơ bộ mà khóa luận thu được đã cho thấy sức mạnh của hệ thống mà khóa luận đã phát triển. Qua đây, khóa luận hi vọng đây sẽ là bàn đạp cho các nghiên cứu liên quan sau này.



# MỤC LỤC

LỜI CẢM ƠN .....	i
ĐỀ CƯƠNG CHI TIẾT .....	ii
TÓM TẮT .....	iv
MỤC LỤC .....	v
DANH MỤC CÁC HÌNH.....	viii
DANH MỤC CÁC BẢNG .....	xi
DANH MỤC THUẬT NGỮ VIẾT TẮT .....	xii
CHƯƠNG 1: GIỚI THIỆU .....	1
1.1 Động lực nghiên cứu: .....	1
1.1.1 Về mặt ứng dụng:.....	1
1.1.2 Về mặt khoa học: .....	2
1.2 Đặt vấn đề và phát biểu bài toán: .....	5
1.2.1 Đặt vấn đề: .....	5
1.2.2 Phát biểu bài toán:.....	6
1.2.3 Các thách thức:.....	8
1.2.4 Đóng góp:.....	8
1.2.4 Bố cục khóa luận:.....	10
Chương 2: TÌNH HÌNH NGHIÊN CỨU VÀ HƯỚNG TIẾP CẬN .....	11
2.1 Giải bài toán SLAM bằng cảm biến độ sâu: .....	11
2.2 Giải bài toán SLAM bằng camera đồng bộ (Stereo): .....	12
2.3 Giải bài toán bằng camera đơn (Monocular):.....	13
2.4 Giải bài toán bằng kết hợp cảm biến một camera và thiết bị đo lường quán tính (IMU):.....	14
2.5 Hướng tiếp cận: .....	15
CHƯƠNG 3: ƯỚC LƯỢNG VỊ TRÍ VÀ TÁI TẠO BẢN ĐỒ LÂN CẬN.....	17
3.1 Phát hiện và theo vết đặc trưng: .....	19
3.1.1 Hệ thống camera đồng bộ (stereo): .....	21

3.2 Khởi động bản đồ lân cận (local map): .....	27
3.3 Xác định vị trí robot: .....	29
3.3.1 Loại nhiễu sử dụng RANSAC: .....	31
3.4 Cập nhật bản đồ lân cận (Mapping): .....	34
3.4.1 Thêm đặc trưng vào bản đồ lân cận: .....	34
3.4.2 Xóa đặc trưng:.....	37
3.5 Phát hiện lỗi trong bản đồ lân cận: .....	37
3.6 Kết hợp với IMU qua lọc nhiễu Kalman UKF (Unscented Kalman filter): .....	38
<b>CHƯƠNG 4: TỐI ƯU ĐỒ THỊ VÀ XÂY DỰNG BẢN ĐỒ</b> .....	40
4.1. Mô hình hóa bài toán SLAM bằng đồ thị:.....	40
4.1.1 Tìm phép chuyển động giữa hai thời điểm: .....	41
4.1.2 Xây dựng đồ thị:.....	42
4.1.3 Phát hiện vòng lặp:.....	43
4.1.4 Tối ưu đồ thị:.....	47
4.2 Xây dựng bản đồ.....	47
<b>CHƯƠNG 5: PHÁT HIỆN VÀ NÉ TRÁNH VẬT CẢN</b> .....	49
5.1 Phát hiện vật cản:.....	49
5.2 Né tránh vật cản:.....	51
5.2.1 Mở rộng kích thước vật cản .....	52
5.2.2 Biểu diễn vật cản.....	53
<b>CHƯƠNG 6: THỰC NGHIỆM</b> .....	58
6.1 Tổng quan hệ thống .....	58
6.1.1 Môi trường.....	58
6.1.2 Thư viện hỗ trợ .....	58
6.1.3 Dữ liệu đầu ra: .....	59
6.2 Thực nghiệm trên bộ dữ liệu stereo của KITTY:.....	59
6.2.1 Giới thiệu về bộ dữ liệu.....	59
6.3 Kết quả thực nghiệm.....	61

CHƯƠNG 7: KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN .....	64
7.1 Tổng kết: .....	64
7.2 Hướng phát triển .....	65
7.2.1 Cải tiến tốc độ xử lý .....	65
7.2.2 Phương pháp tối ưu toàn cục:.....	65
DANH MỤC TÀI LIỆU THAM KHẢO.....	66

# DANH MỤC CÁC HÌNH

Hình 1-1 Một số ứng dụng của SLAM trong thực tế: a) Drone thả áo phao cứu hộ ở những nơi nguy hiểm khó tiếp cận với con người, b) Dự án PrimeAir giao hàng bằng máy bay của Amazon.

Hình 1-2 Một số ứng dụng hỗ trợ cho con người: a) Drone dẫn đường cho người khiếm thị, b) Quay phim, chụp ảnh với drone

Hình 1 3 a) Thiết bị Microsoft Kinect b) Camera đồng bộ (Stereo)

Hình 1-4 Mô hình điều hướng của robot tự hành

Hình 2-1 Mô hình xử lý của hệ thống định vị và tái tạo bản đồ bằng hai camera đồng bộ kết hợp thiết bị đo lường quán tính (IMU).

Hình 3-1 Minh họa cho bài toán ước lượng chuyển động bằng thông tin thị giác. Điểm màu xanh là điểm trong không gian ba chiều trên thực tế. Điểm màu đỏ là tọa độ trên ảnh của điểm thực tế khi camera quan sát được tại hai thời điểm khác nhau  $t, t'$ .

Hình 3-2 Phát hiện và theo vết đặc trưng của hai camera từng thời điểm

Hình 3-3 Các điểm màu xanh là đặc trưng góc Shi-Tomasi tại camera chính và phụ cùng thời điểm  $t$ . Điều kiện ràng buộc để thỏa mãn cặp tương đồng epipolar là một điểm phải nằm trên đường thẳng epipolar line tương ứng với điểm tương đồng.  $I_{2p1}$  nằm trên đường thẳng epipolar  $L1'$  (màu cam) tương ứng với  $I_{p1}$  nên  $I_{p1}$  và  $I_{2p1}$  thỏa mãn điều kiện cặp tương đồng. Trong khi đó  $I_{2p2}$  không nằm trên  $L2'$  là epipolar tương ứng của  $I_{p2}$  nên không phải là cặp tương đồng.

Hình 3-4 So khớp giữa hai ảnh từ camera chính và phụ cùng thời điểm. Bên trái là ảnh từ camera chính, bên phải là ảnh camera phụ. Các đường nối vị trí của các cặp tương ứng.

Hình 3-5 Các cặp đặc trưng sau khi đã loại nhiều trong việc so khớp

Hình 3-6 Theo vết đặc trưng theo từng khung hình đến từ camera chính

Hình 3.7 Minh họa cho quan sát của đặc trưng  $w_{p_i}$  tại hai thời điểm  $t, t'$  trên mặt phẳng ảnh camera chính. Vector màu xanh  $\vec{k}_{it}$  và  $\vec{k}_{it'}$  lần lượt là quan sát của  $p_i$  tại  $t, t'$ . Các vector trên có hướng từ vị trí robot từng thời điểm ( $r_t = O_{(C1)_t}$ ) chỉ đến vị trí

đặc trưng  $p_i$  trong tọa độ thế giới. Vector quan sát của đặc trưng có thể tính từ vị trí đặc trưng trên mặt phẳng ảnh camera chính.

Hình 3-8 a) Minh họa cho phép chiếu  $K$  chiếu một điểm từ hệ tọa độ camera (C)  ${}^Cp_i$  xuống mặt phẳng ảnh.

b) Minh họa cho phép chiếu  $K^{-1}$  chiếu một điểm trên mặt phẳng ảnh thành vector có hướng từ tâm camera  $O_{(C)t}$  qua điểm 3D của nó ở tọa độ camera  ${}^Cp_i$ .

Hình 3-9 a) Minh họa phép xoay từ hệ trục camera sang hệ trục thế giới.

b) Khi áp dụng phép xoay vào, vector quan sát  $\vec{k}_{it}^C$  trở thành  $\vec{k}_{it}$  và có hướng từ tâm camera qua vị trí 3D của đặc trưng thứ  $i$  trong hệ tọa độ thế giới

Hình 3-10 Minh họa các vị trí camera thời điểm  $t$  và sai số so với vector quan sát  $\vec{k}_{it}$ . Vị trí có sai số nhỏ nhất (tổng diện tích hình bình hành đỏ nhỏ nhất) sẽ là vị trí robot thời điểm  $t$ . Ở đây là vị trí  $r_i$  có sai số nhỏ nhất so với vector quan sát được.

Hình 3-11 Minh họa thuật toán RANSAC để khớp một đường thẳng lên tập dữ liệu hai chiều. (a) Đầu tiên,  $k$  điểm được chọn ngẫu nhiên từ tập dữ liệu ( $k = 2$ ). (b) Từ những điểm này, ta sẽ cho ra được một mô hình. Ở đây, một đường thẳng được vẽ qua hai điểm và các điểm nằm trong vùng màu xanh sẽ được xem là phù hợp với mô hình. (c) Với mỗi điểm trong tập dữ liệu ta sẽ tìm sai số của điểm đó so với mô hình (khoảng cách từ điểm đến đường thẳng). Nếu sai số này là đủ nhỏ thì ta xem các điểm này là phù hợp với mô hình (điểm màu xanh) và loại bỏ các điểm không phù hợp (điểm màu đỏ). Mô hình cuối cùng sẽ được tính lại chỉ dựa vào các điểm màu xanh.

Hình 3-12 Minh họa các vị trí đặc trưng  $i$  thời điểm 1, 2 và sai số so với vector quan sát  $\vec{k}_{i1}, \vec{k}_{i2}$ . Vị trí có sai số nhỏ nhất (tổng diện tích hình bình hành đỏ nhỏ nhất) sẽ là vị trí đặc trưng ở tọa độ thế giới. Ở đây là vị trí  ${}^w p_1$  có sai số nhỏ nhất so với vector quan sát được.

Hình 4-1 Quá trình xây dựng và tối ưu đồ thị

Hình 4-2 Đám mây điểm ban đầu và đám mây điểm sau khi đã sử dụng bộ lọc lưới voxel (phải). Tuy kích thước của đám mây điểm đã giảm nhưng vẫn thể hiện rõ môi trường.

Hình 5-1 Các điểm màu đỏ là các đặc trưng đã có vị trí 3D xác định bởi hệ thống một camera, ta có thể thấy các đặc trưng tập trung chủ yếu tại các vật thể (xe cộ, cây cối trên đường và ít tập trung tại mặt đường và hoàn toàn không có trong không khí)

Hình 5-2 Minh họa thuật toán k-means, với điểm trọng tâm là tam giác. Lần lặp đầu tiên sẽ random ngẫu nhiên k điểm trọng tâm, các điểm còn lại sẽ xếp vào nhóm của điểm trọng tâm có khoảng cách gần nhất, sau đó thuật toán sẽ tính lại các điểm trọng tâm. Vòng lặp tiếp theo cũng sẽ thực hiện các bước tương tự cho đến khi các điểm trọng tâm không còn thay đổi.

Hình 5-3 Ví dụ về khung bao 3D bao quanh vật cản đã phát hiện. Vị trí vật cản (tâm khung bao) đã biết trong pha phát hiện vật cản và vị trí robot (ở đây là chiếc xe màu trắng) đã có trong pha SLAM.

Hình 5-4 Robot (tròn) và vật cản ( chữ nhật )

Hình 5-5 Robot dạng điểm và vật cản sau khi mở rộng kích thước

Hình 5-6 Biên chuyển động đồng thời cũng là biên quan sát thiết bị

Hình 5-7 Góc biểu diễn vật cản  $\phi_{obs1_l}$  và  $\phi_{obs1_r}$  là góc biên trái và phải của vật cản thứ 1

Hình 5-8 Danh sách khoảng trống

Hình 5-9 Đường trung tuyến của khoảng trống

Hình 5-10 Đường đi cho robot né tránh vật cản và hướng đến đích

Hình 6-1 Hình ảnh từ camera đồng bộ bộ dữ liệu KITTY, bên trên là ảnh từ camera chính, bên dưới là ảnh từ camera phụ cùng một thời điểm

Hình 6-2 Chạy chương trình trên bộ dữ liệu offline

# DANH MỤC CÁC BẢNG

Bảng 6-1 Kết quả chạy chương trình

# DANH MỤC THUẬT NGỮ VIẾT TẮT

<b>EKF</b>	Extended Kalman Filter – Bộ lọc Kalman mở rộng
<b>ORB</b>	Oriented FAST and Rotated BRIEF – FAST có hướng và BRIEF xoay
<b>PHD-F</b>	Probability Hypothesis Density Filter - Bộ lọc phân bố xác suất
<b>RANSAC</b>	RANdom SAmple Consensus – Tập mẫu ngẫu nhiên đồng thuận
<b>SLAM</b>	Simultaneous Localization And Mapping - Định vị và xây dựng bản đồ đồng thời
<b>SURF</b>	Speeded Up Robust Feature – Đặc trưng bền vững tốc độ nhanh
<b>KLT</b>	Kanade–Lucas–Tomasi feature tracker – phương pháp theo vết Kanade-Lucas-Tomasi
<b>IMU</b>	Inertial Measurement Unit – Thiết bị đo lường quán tính
<b>GPS</b>	Global Positioning System - Hệ thống định vị toàn cầu



# CHƯƠNG 1: GIỚI THIỆU

*Visual SLAM là một bài toán hiện đang nhận được sự quan tâm rất lớn trong cộng đồng thị giác máy tính vì khả năng ứng dụng của nó. Chương này sẽ tập trung trình bày về động lực nghiên cứu và giới thiệu sơ lược về bài toán Visual SLAM. Bố cục của khóa luận cũng sẽ được trình bày ở cuối chương.*

## **1.1 Động lực nghiên cứu:**

### **1.1.1 Về mặt ứng dụng:**

Ngày nay, những ứng dụng của robot tự hành ngày càng nhiều, góp phần vào cuộc sống của con người với những khả năng hữu ích của mình. Sự phổ biến của robot tự động ngày càng lớn trong mọi lĩnh vực có thể kể đến như: y tế, thương mại, truyền thông, thám hiểm, du lịch, ...

Có thể hiểu robot tự hành như một cỗ máy có thể tiếp nhận và xử lý các thông tin từ môi trường bên ngoài đồng thời phản ứng lại mà không cần sự điều khiển của con người nhằm thuận tiện tối đa cho người dùng. Ví dụ như robot có khả năng biết được vật cản trước mặt và né tránh đồng thời đến đích nhằm thực hiện nhiệm vụ, ...

Ứng dụng của robot tự hành đã xuất hiện rộng khắp, có thể kể đến như: máy bay không người lái (drone) cứu hộ mang đến cho con người những dụng cụ y tế cần thiết trong thời gian nhanh nhất khi gặp tình huống nguy cấp, drone hỗ trợ trong công tác tìm người lạc từ trên cao, drone trợ giúp lính cứu hỏa khi mang lại thông tin cục bộ về đám cháy, drone trong những dịch vụ chuyển phát nhanh như giao hàng của hãng Amazon, drone thám hiểm, ...



(a)



(b)

Hình 1-1 Một số ứng dụng của SLAM trong thực tế: a) Drone thả áo phao cứu hộ ở những nơi nguy hiểm khó tiếp cận với con người, b) Dự án PrimeAir giao hàng bằng máy bay của Amazon.

Cùng với những ứng dụng kể trên, với sự phát triển mạnh mẽ của trí tuệ nhân tạo và việc giá cả robot (drone) tự hành ngày càng giảm đã kéo theo những xu hướng mới về cá nhân hóa drone để hỗ trợ con người trong cuộc sống thường ngày. Ứng dụng đã có hoặc đang phát triển có thể kể đến như: dẫn đường cho người khiếm thị, trông trẻ từ xa, drone để phục vụ nhu cầu chụp ảnh, quay phim trên cao, ...



(a)



(b)

Hình 1-2 Một số ứng dụng hỗ trợ cho con người: a) Drone dẫn đường cho người khiếm thị, b) Quay phim, chụp ảnh với drone

### 1.1.2 Về mặt khoa học:

Nhu cầu về robot tự hành ngày càng lớn kéo theo đó là sự phát triển của những bài toán con về ước lượng vị trí xây dựng bản đồ (SLAM), dò tìm đường, né tránh vật cản, ... Việc robot tự hành được thì robot cần biết vị trí hiện tại trong không gian (ước lượng vị trí) đồng thời biết được môi trường xung quanh (xây dựng bản đồ), vấn đề này (SLAM) là cốt lõi trong hệ thống robot tự hành. Sau đó với các thông tin vị trí

và bản đồ có được, robot sẽ dò tìm đường đi đồng thời với việc phát hiện các vật cản trên đường đi và né tránh để đến đích. Để làm được điều đó, robot cũng cần một hệ thống điều hướng phù hợp, linh hoạt đồng thời cũng phải tính toán đến các vấn đề tiết kiệm chi phí, năng lượng và có khả năng tự hành cao.

Có thể nói, bài toán SLAM về cơ bản đã được giải quyết. Tuy nhiên, khi áp dụng vào thực tế thì còn rất nhiều vấn đề cần giải quyết. Các vấn đề phát sinh chủ yếu từ nhiều của thiết bị và yêu cầu xử lý trong thời gian thực mà một hệ thống điều hướng cần phải đáp ứng.

Nếu robot ở ngoài trời, vị trí robot có thể có được qua hệ thống GPS. Tuy nhiên tính hiệu GPS là rất nhiều và không phải lúc nào cũng thể có được, đặc biệt là ở những môi trường như trong nhà, dưới biển, ... Vì thế, các hệ thống SLAM chủ yếu định vị vị trí dựa trên những thiết bị được trang bị trên robot như camera, laser, bộ thu phát sóng siêu âm, ... Các cảm biến thường được sử dụng trên SLAM hiện nay là:

- **Máy quét la-de:** có thể cho ra dữ liệu độ sâu với độ chính xác cao. Tuy nhiên thiết bị có tốc độ quét chậm, khó đáp ứng nhu cầu phản ứng nhanh của SLAM đồng thời có nhược điểm giá thành cao và trọng lượng nặng nên cũng không thể trang bị cho hệ thống SLAM.
- **Bộ thu phát sóng siêu âm (sonar):** có nguyên lý giống như loài dơi, cũng có thể cho ra dữ liệu độ sâu và với tốc độ cao. Nhưng hạn chế về mặt dữ liệu rất khó hiểu với con người và khó đạt được kết quả chính xác. Thiết bị thường được sử dụng để xác định và né tránh vật cản.
- **Thiết bị đo lường quán tính (IMU):** thiết bị cho biết gia tốc, góc xoay theo các chiều với tần số rất cao (trên 100Hz). Đồng thời giá thành IMU cũng rất thấp, tuy nhiên những đo lường của IMU có rất nhiều nhiễu và kết quả tính toán được dễ dàng bị trôi (drift) theo thời gian. Vì thế, IMU thường được sử dụng để kết hợp với các cảm biến khác trong các hệ thống SLAM.

- **Camera:** Ngoài các thiết bị kể trên, còn một hướng nghiên cứu đó là sử dụng camera để làm cảm biến. Hướng nghiên cứu này được gọi là Visual SLAM, và đang phát triển mạnh mẽ trong cộng đồng thị giác máy tính vì khả năng thu dữ liệu nhanh và nhiều thông tin là hình ảnh của Camera đồng thời với giá thành rẻ, gọn nhẹ nên Camera là lựa chọn hàng đầu trong các hệ thống SLAM.



(a)



(b)

*Hình 1-3 a) Thiết bị Microsoft Kinect b) Camera đồng bộ (Stereo)*

Nhận thấy những ưu điểm trong việc sử dụng Camera làm thiết bị cảm biến cho hệ thống SLAM. Nên khóa luận sẽ xem Visual SLAM là hướng nghiên cứu chính để tìm hiểu và phát triển. Đồng thời hệ thống cũng kết hợp thêm dữ liệu từ IMU để đáp ứng nhu cầu phản ứng nhanh của robot mà tần số từ thiết bị của Camera không thể đáp ứng (khoảng 25-60Hz). Tuy nhiên việc gắn Camera cũng có nhiều vấn đề quan tâm:

- **Hệ thống một camera (Monocular):** hệ thống SLAM chỉ có duy nhất một Camera làm thiết bị cảm biến đang được cộng đồng thế giới quan tâm vì tính đơn giản và vì những thiết bị điện thoại hiện có rất nhiều trên thị trường. Tuy nhiên, hệ thống monocular khó để khôi phục dữ liệu độ sâu, vì thế kéo theo đó là ít hệ thống có khả năng sử lý trong thời gian thực.
- **Hệ thống Camera độ sâu (Kinect):** hệ thống khắc phục việc thiếu dữ liệu độ sâu bằng cảm biến hồng ngoại, tiêu biểu là thiết bị Microsoft Kinect. Tuy nhiên thiết bị có điểm yếu là khó sử dụng được ngoài trời (outdoor) vì cảm biến hồng ngoại nhiễu rất nhiều với ánh sáng đồng thời chỉ phản ánh được độ sâu của

những vật thể ở rất gần (khoảng giữa từ 0.5m-4m) là không đủ để cho hệ thống có thể hoạt động tự động.

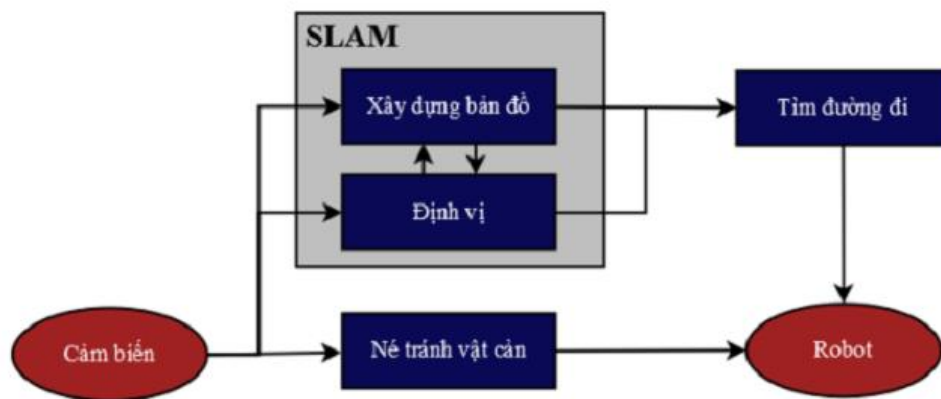
- **Hệ thống Camera đồng bộ (Stereo):** hệ thống gồm hai camera có góc nhìn trùng lặp nhau. Hệ thống có thể tái tạo lại độ sâu của ảnh trong nhà lẫn ngoài trời, dữ liệu độ sâu có được giúp hệ thống có thể phản ứng nhanh với môi trường đồng thời xác định vị trí và tái tạo bản đồ chính xác.

Vì những ưu điểm kể trên của hệ thống Camera đồng bộ (Stereo), nên khóa luận nghiên cứu theo hướng sử dụng hệ thống gồm hai camera trong bài toán SLAM kết hợp với IMU. Những hạn chế kể trên của hệ thống SLAM có thể được khắc phục bằng việc sử dụng hệ thống kết hợp Camera đồng bộ và IMU.

## 1.2 Đặt vấn đề và phát biểu bài toán:

### 1.2.1 Đặt vấn đề:

Các nhiệm vụ trong điều hướng robot di động tự hành có thể mô hình hoá theo luận văn [5] như sau:



Hình 1-4 Mô hình điều hướng của robot tự hành

- **Xây dựng bản đồ (Mapping):** tái tạo lại bản đồ của môi trường dựa trên quan sát và thông tin vị trí của robot. Đồng thời thông tin bản đồ giúp ước lượng vị trí của robot ở những thời điểm khác.
- **Định vị (Localization):** xác định vị trí hiện tại của robot trong môi trường dựa vào quan sát hiện tại và bản đồ đã xây dựng trước đó.

- Né tránh vật cản (Obstacle Detection and Avoidance): nếu muốn robot có khả năng tự hành, robot phải có khả năng phát hiện và tránh né được các vật cản trên đường di chuyển.
- Tìm đường đi (Path Planning): khi đã xây dựng bản đồ và có vị trí hiện tại, robot sẽ tìm đường đi đến đích đã cho trước.

Hai bài toán đầu đặc biệt quan trọng và luôn cần phải được giải quyết trong các hệ thống điều hướng tự động. Chính vì vậy, khóa luận sẽ tập trung vào giải quyết hai bài toán là định vị và xây dựng bản đồ với thiết bị camera đồng bộ (Stereo) và IMU. Ngoài ra còn khóa luận còn trình bày thêm một phần về hai bài toán con tiếp theo là né tránh vật cản và tìm đường đi. Mục tiêu là một hệ thống tự hành hoàn toàn có khả năng hoạt động ở nhiều môi trường cả trong nhà lẫn ngoài trời.

### **1.2.2 Phát biểu bài toán:**

a/ Nhiệm vụ:

- Khóa luận tập trung vào việc nghiên cứu về định vị và tái tạo bản đồ từ dữ liệu hình ảnh và IMU ở môi trường trong nhà lẫn ngoài trời mà không biết trước bản đồ.
- Khóa luận có tính đến vấn đề phát hiện và né tránh vật cản dựa vào bản đồ thu nhận được từ việc định vị và tái tạo bản đồ.
- Từ việc định vị và tái tạo bản đồ cùng với phát hiện, né tránh vật cản, luận văn xây dựng một hệ thống điều hướng hoàn chỉnh, giúp cho robot có khả năng định hướng về đích với tọa độ cho trước lúc khởi động.

b/ Phát biểu:

Về cơ bản, hệ thống SLAM có thể định nghĩa như sau. Gọi  $P_t$  là vị trí của robot so với hệ trục tọa độ thế giới WCS tại thời điểm  $t$  (tương đương với hệ tọa độ hiện tại của camera chính trong hệ thống Stereo) được biểu diễn bằng một ma trận  $4 \times 4$ .

$$P_t = \begin{bmatrix} & & T_x \\ & R & T_y \\ & & T_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (1.1)$$

Trong đó:

$$R_{(3 \times 3)} = \begin{bmatrix} \cos \theta \cos \psi & \cos \phi \sin \psi + \sin \phi \sin \theta \cos \psi & \sin \phi \sin \psi - \cos \phi \sin \theta \cos \psi \\ -\cos \theta \sin \psi & \cos \phi \cos \psi - \sin \phi \sin \theta \sin \psi & \sin \phi \cos \psi + \cos \phi \sin \theta \sin \psi \\ \sin \theta & -\sin \phi \cos \theta & \cos \phi \cos \theta \end{bmatrix}$$

là ma trận biểu diễn cho phép xoay với  $\phi$ ,  $\theta$ ,  $\psi$  là góc quay theo trục x, y, z và  $T = (T_x, T_y, T_z)$  là vector biểu diễn cho phép tịnh tiến. Vị trí robot phụ thuộc vào 6 biến: 3 biến biểu diễn phép tịnh tiến và 3 biến biểu diễn cho phép xoay, nên thường được gọi là có 6 bậc tự do (6-DOF). Có thể coi vị trí của robot chính là phép biến đổi từ hệ trục thế giới sang hệ trục tọa độ hiện tại của camera bao gồm phép xoay R và phép tịnh tiến T. Vị trí robot tính theo hệ trục thế giới WCS cũng chính là  $T = (T_x, T_y, T_z)$  và R là hướng của robot trong hệ trục thế giới.

Ngoài ra, tại mỗi thời điểm, robot còn nhận được quan sát từ cảm biến, kí hiệu là  $O_k, I_k$ . Mỗi quan sát này sẽ bao gồm ảnh từ camera đồng bộ  $O_k$  có kích thước là 640x480 và các thông tin từ IMU là  $I_k$ . Nhiệm vụ của bài toán SLAM là ước lượng vị trí  $P_k$  từng thời điểm, đồng thời xây dựng bản đồ m mà chỉ dựa vào các quan sát  $\{O_1, O_2, O_3, \dots, O_k\}$  và  $\{I_1, I_2, I_3, \dots, I_k\}$  và vị trí ban đầu  $P_0$ . Tuy nhiên, đối với  $P_0$  ta có thể chọn tùy ý mà không ảnh hưởng đến kết quả nên khóa luận sẽ chọn  $P_0$  làm gốc tọa độ.

$$P_0 = P_w = I_4 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (1.2)$$

### 1.2.3 Các thách thức:

Đối với bài toán SLAM sử dụng camera đồng bộ và IMU mà ta đang xem xét, ngoài những thách thức đang tồn đọng từ bài toán SLAM, còn có những thách thức đến từ cảm biến được sử dụng. Các thách thức này chính là:

- **Xử lý nhiễu.** Dữ liệu ảnh từ camera gặp hai vấn đề chính đó là nhiễu và bị mất mát thông tin. Để tăng độ chính xác của hệ thống thì quá trình xử lý nhiễu phải đồng thời giải quyết được hai vấn đề này. Ngoài ra, nhiễu còn đến từ chính quá trình định vị, một hệ thống SLAM tốt cần phải đảm bảo độ chính xác trong vị trí ước lượng được, tránh để sai số quá lớn so với đường đi thực tế.
- **Vật cản.** Thách thức đến từ việc né tránh vật cản đến từ việc robot phải tìm ra được cách di chuyển để né tránh vật cản.
- **Tốc độ xử lý.** Tuy sức mạnh của máy tính đã được tăng cường đáng kể trong những năm gần đây, việc một hệ thống SLAM có thể chạy được trong thời gian thực (30 – 60 Hz) vẫn luôn là một thách thức lớn.
- **Môi trường động.** Hiện nay, các nghiên cứu về SLAM vẫn chủ yếu được thực hiện trên các môi trường tĩnh, ít thay đổi. Tuy nhiên, trên thực tế, môi trường bên ngoài là luôn luôn biến động. Một hệ thống SLAM có khả năng xử lý những sự biến đổi trong môi trường sẽ là một bước đột phá lớn trong lĩnh vực khoa học robot.

### 1.2.4 Đóng góp:

Trong những vấn đề khóa luận vừa đưa ra, khóa luận sẽ cố gắng giải quyết hai thách thức lớn là xử lý nhiễu và tốc độ tính toán của hệ thống. Mục tiêu cuối cùng của khóa luận thông qua việc giải quyết các khó khăn này là đưa ra được một hệ thống SLAM tự động có khả năng hoạt động trong thời gian thực. Những đóng góp chính của khóa luận trong quá trình giải quyết các thách thức trên có thể được chia ra thành hai phần là lý thuyết và thực tiễn:



### Về mặt lý thuyết:

- Khóa luận đã tìm hiểu được một phương pháp để kết hợp thông tin từ hình ảnh và IMU để định vị và tái tạo bản đồ. Với phương pháp này, kết quả vị trí của robot có thể được tính toán với tần số cao (100Hz) đáp ứng được nhu cầu phản ứng nhanh để điều khiển của robot tự hành.
- Khóa luận đã dùng phương pháp kết hợp từ hai camera đồng bộ để vượt qua khuyết điểm của camera độ sâu, giúp hệ thống có thể tự động vận hành trong nhiều môi trường khác nhau (trong nhà, ngoài trời, ...).
- Khóa luận đã trình bày phương pháp phát hiện vật cản dựa vào thông tin thị giác và chọn ra được một phương pháp phù hợp cung cấp thông tin về vật cản trong môi trường đầy đủ với sai số thấp.
- Khóa luận đã trình bày các phương pháp né tránh vật cản nổi trội và thông dụng hiện nay và áp dụng được một phương pháp né tránh vật cản có độ an toàn cao vào mô hình để đưa robot tới đích.
- Khóa luận cũng đã đưa ra được quy trình cho các hệ thống SLAM có xét yếu tố vật cản sử dụng thông tin từ camera và IMU. Kết hợp nghiên cứu về định vị, tái tạo bản đồ và phát hiện, né tránh vật cản để xây dựng một hệ thống SLAM đầy đủ.

### Về mặt thực tiễn:

- Khóa luận đã xây dựng được một ứng dụng hoàn chỉnh có khả năng tái tạo lại bản đồ ba chiều và đường đi của rô-bốt có thể nhận dữ liệu từ hai camera và IMU có sẵn. Ứng dụng có khả năng chạy được trong thời gian thực. Điều này thể hiện tính hiệu quả và tốc độ của hệ thống mà khóa luận đã xây dựng.
- Khóa luận cũng chỉ ra tiềm năng của mô hình có thể phát triển các ứng dụng thực tế như hỗ trợ người khiếm thị di chuyển, vận chuyển hàng hóa, giám sát từ xa, phục vụ nhu cầu chụp ảnh, quay phim ngoài trời, ...

### 1.2.4 Bố cục khóa luận:

Nội dung khóa luận trình bày gồm có 7 chương:

**Chương 1: Giới thiệu.** Trình bày về động lực nghiên cứu cũng như thách thức của bài toán đồng thời với các đóng góp của khóa luận.

**Chương 2: Tình hình nghiên cứu và Hướng tiếp cận.** Chương này sẽ đi qua các hướng tiếp cận chính để giải quyết bài toán SLAM. Ngoài ra, hướng tiếp cận mà khóa luận chọn cũng sẽ được trình bày trong chương này.

**Chương 3: Ước Lượng Vị Trí Và Tái Tạo Bản Đồ Lân Cận.** Ở chương này sẽ trình bày về giải thuật SLAM, đồng thời xác định vị trí và tái tạo bản đồ lân cận.

**Chương 4: Tối Ưu Đồ Thị Và Xây Dựng Bản Đồ.** Chương này sẽ trình bày cách mà hệ thống SLAM được tối ưu thông qua việc mô hình như một đồ thị. Đồng thời tái tạo toàn bộ bản đồ 3D mà robot quan sát được trên đường đi

**Chương 5: Phát Hiện Và Né Tránh Vật Cản.** Sau khi đã có vị trí và bản đồ trong khâu SLAM, hệ thống sẽ giúp robot phân biệt được các vật cản và tiến hành né tránh đi đến đích đã cho.

**Chương 6: Thực Nghiệm.** Chương này sẽ trình bày chương trình khóa luận cài đặt cùng với các kết quả thực nghiệm thu được từ một bộ dữ liệu offline.

**Chương 7: Kết Luận Và Hướng Phát Triển.** Chương này sẽ tổng kết kết quả của khóa luận. Ngoài ra cũng sẽ bàn đến những hướng phát triển trong tương lai.

## Chương 2: TÌNH HÌNH NGHIÊN CỨU VÀ HƯỚNG TIẾP CẬN

*Chương này sẽ tập trung trình bày về tình hình nghiên cứu về SLAM hiện nay và một số nghiên cứu về phát hiện, né tránh vật cản. Khóa luận sẽ so sánh các hướng tiếp cận phổ biến nhất hiện nay đó là sử dụng các dạng camera khác nhau. Như chương trước đã đề cập, khóa luận sẽ tập trung vào các nghiên cứu Visual SLAM. Phần cuối chương sẽ đề cập đến hướng tiếp cận mà khóa luận sẽ chọn để giải quyết bài toán đã đề ra.*

### 2.1 Giải bài toán SLAM bằng cảm biến độ sâu:

Như đã trình bày, các loại cảm biến độ sâu thường có hai loại chính là máy quét laser và camera độ sâu (Microsoft Kinect). Ý tưởng chính của phương pháp dùng cảm biến độ sâu là sử dụng những bản đồ độ sâu để so khớp (3D-to-3D) để tìm chuyển động giữa hai thời điểm liên tiếp, và với bản đồ 3D môi trường xung quanh, robot có thể dễ dàng di chuyển trong môi trường đồng thời né tránh các vật cản.

Các hệ thống SLAM tự động sử dụng cảm biến độ sâu như thiết bị cảm ứng bên ngoài. Những cặp dữ liệu thu nhận được được sử dụng để ước lượng chuyển động của hệ thống. Một số nghiên cứu nổi bật như sau:

- [16] sử dụng phương pháp sử dụng thuật toán lặp tìm điểm gần nhất (Iterative Closest Point - ICP), tuy nhiên ICP có khả năng hội tụ thấp với những thay đổi lớn trong dữ liệu.
- [15], [14] đề xuất phương pháp scan với nhiều độ phân giải tương quan lẫn nhau cho ra kết quả rất chính xác.
- [3], [20] rút trích các đặc trưng như góc, đường thẳng và dùng để liên kết dữ liệu giữa hai thời điểm. Với cách rút trích này, hệ thống tiết kiệm được thời gian tính toán rất lớn.

Sau đây là đánh giá chung của các hệ thống SLAM sử dụng cảm biến độ sâu:

- Ưu điểm: chính xác, đơn giản và có bản đồ hỗ trợ cho các khâu di chuyển.
- Khuyết điểm: đối với laser: có trọng lượng lớn, khó kết hợp vào robot cần tốc độ nhanh; giá thành cao; đối với camera độ sâu: dễ dàng nhiễu với môi trường nhiều ánh sáng; khoảng đo được độ sâu thấp. Ngoài ra chi phí tính toán cũng là vấn đề với SLAM sử dụng cảm biến độ sâu cho hệ thống có thể hoạt động với thời gian thực; ước lượng giữa các thời điểm với nhau nên kết quả dễ trôi, cần những thuật toán tối ưu toàn cục không chế.

## 2.2 Giải bài toán SLAM bằng camera đồng bộ (Stereo):

Từ camera đồng độ, hệ thống có thể khôi phục được độ sâu của ảnh và dùng độ sâu đó để tính toán vị trí đồng thời cập nhật bản đồ. Một số nghiên cứu nổi bật có thể kể đến:

- [13] Nister đã sử dụng hệ thống camera đồng bộ, dựa trên việc rút trích đặc trưng góc và so khớp giữa hai camera để có vị trí 3D của các đặc trưng. Sau đó so khớp đặc trưng giữa hai ảnh chính (key frame) liên tiếp nhau ở camera chính và tìm mối liên hệ giữa vị trí 3D và vị trí 2D trên ảnh (3D-to-2D) để ước lượng được chuyển động của camera và sử dụng RANSAC để loại nhiễu trong khâu này. Tuy nhiên, do việc tính toán lớn nên hệ thống chưa áp dụng vào thời gian thực.
- [4] cũng sử dụng phương pháp phát hiện và theo vết đặc trưng trên mỗi khung ảnh của cả hai camera, tuy nhiên chuyển động được ước lượng bởi vị trí 2D tại bốn ảnh (hai ảnh từ hai camera ở hai thời điểm liên tiếp) (2D-to-2D) cho kết quả chính xác hơn so với (3D-to-2D) truyền thống và có thể chạy thực tế. Tuy nhiên, kết quả bị ảnh hưởng khá nhiều bởi góc xoay của robot.

Đánh giá chung ưu và khuyết của hệ thống SLAM bằng camera đồng bộ:

- Ưu điểm: có độ sâu được từ camera đồng bộ, có thể sử dụng ngoài trời.
- Khuyết điểm: phụ thuộc vào việc rút trích đặc trưng, nếu môi trường đồng nhất (ít đặc trưng) thì hệ thống dễ thất bại; việc tìm độ sâu cũng bị ảnh hưởng rất

lớn khi độ sâu của môi trường lớn hơn độ sâu của khoảng cách giữa hai camera (baseline); việc rút trích đặc trưng để so khớp rất ảnh hưởng khả năng tính toán.

## 2.3 Giải bài toán bằng camera đơn (Monocular):

Sự khác biệt của hệ thống camera đơn với các hệ thống khác là cả chuyển động tương quan và bản đồ 3D đều phải được tính toán từ dữ liệu ảnh 2D hoàn toàn. Vì khoảng cách chính xác là không biết, nên các hệ thống thường gán khoảng cách giữa hai vị trí đầu tiên là một và được xác định sau đó. Có ba hướng chính trong hệ thống một camera đơn: dựa vào đặc trưng (feature-based), dựa vào giá trị pixel của ảnh (appearance-based) và phương pháp lai giữa hai phương pháp trên. Trong khi phương pháp dựa vào đặc trưng tìm kiếm và theo vết các đặc trưng tĩnh trong ảnh, phương pháp dựa vào sự giá trị tất cả pixel trong ảnh và phương pháp lai sử dụng kết hợp hai phương pháp trên. Các nghiên cứu nổi bật:

- Hệ thống SLAM một camera đơn thời gian thực trong không gian rộng lớn (large-scale) đầu tiên được [13] giới thiệu. Nghiên cứu sử dụng RANSAC để loại nhiễu và 3D-to-2D để tính toán vị trí camera mới. Tiếp nối đó, [12] giới thiệu phương pháp năm điểm (five-point) để tính giả thuyết vị trí cho RANSAC và trở nên rất phổ biến để giải quyết bài toán SLAM với một camera.
- [5] theo hướng tiếp cận sử dụng một camera đa hướng kết hợp với optical flow. [9] và [11] sử dụng một cửa sổ điều chỉnh lân cận (local windowed-bundle adjustment) để tối ưu cục bộ kết quả chuyển động và bản đồ.
- Một hệ thống SLAM một camera được áp dụng cho quãng đường xe chạy (2.5 km) [3]. Ở nghiên cứu này đã chia nhỏ ước lượng chuyển động thành ước lượng phép xoay và phép dịch chuyển. Phép xoay tính bằng những điểm ở vô cùng, phép dịch chuyển từ việc khôi phục bản đồ 3D. Những đặc trưng so khớp sai loại bằng phương pháp năm điểm với RANSAC.

Đánh giá chung hệ thống SLAM một camera đơn:

- Ưu điểm: hệ thống một camera đơn giản, có thể phát triển rộng rãi khi điện thoại di động thường chỉ có một camera. Đã có những hệ thống hoạt động ở môi trường ngoài trời và với quãng đường rất dài trong thời gian thực.
- Khuyết điểm: hệ thống không biết được khoảng cách chính xác, chỉ biết được tỷ lệ tương quan giữa các khoảng cách. Việc tính toán thường phức tạp hơn so với các hệ thống khác do phải xử lý từ ảnh 2D. Hướng tiếp cận dựa vào đặc trưng đòi hỏi môi trường nhiều đặc trưng và môi trường cũng phải tĩnh. Hướng tiếp cận dựa vào giá trị pixel của ảnh dễ nhiều khi bị che khuất và đòi hỏi độ phức tạp tính toán lớn hơn tiếp cận bằng đặc trưng.

## **2.4 Giải bài toán bằng kết hợp cảm biến một camera và thiết bị đo lường quán tính (IMU):**

Vì một camera có nhược điểm không biết được quãng đường chính xác, một số nghiên cứu đã kết hợp một camera và thiết bị đo lường quán tính để khôi phục quãng đường chính xác. Khoảng cách tính theo đơn vị mét (m) được ước lượng từ thông tin gia tốc từ IMU. Ngoài ra, kết quả đầu ra được kết hợp với thông tin có tần số cao từ IMU thông qua bộ lọc Kalman nhằm tăng tần số kết quả đầu ra cuối cùng, việc này rất quan trọng để đáp ứng với hệ thống robot cần tốc độ phản ứng cao. Một số nghiên cứu nổi bật [2] [3].

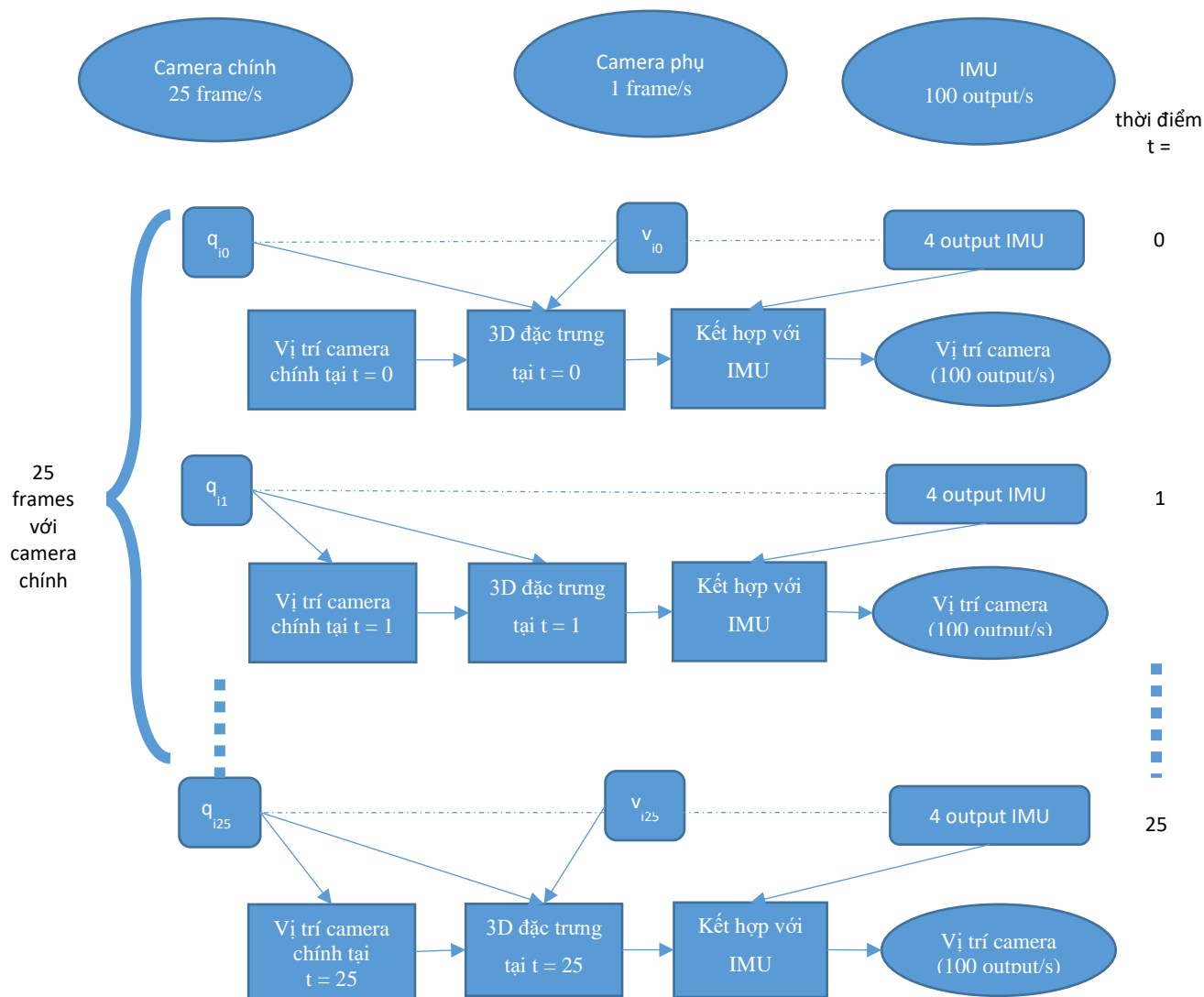
Đánh giá chung:

- Ưu điểm: kết hợp thông tin từ IMU để khắc phục khuyết điểm một camera là khoảng cách chính xác; thông tin đầu ra với tần số cao hơn và chính xác hơn do kết hợp từ hai cảm biến.
- Khuyết điểm: khó kết hợp được với các robot khi cần thông tin gia tốc khác không, nhiều trường hợp di chuyển với tốc độ cố định hoặc ở trạng thái lơ lửng đối với drone.

## 2.5 Hướng tiếp cận:

Qua phần trình bày vừa rồi, khóa luận đã khảo sát các hướng tiếp cận chính của hệ thống định vị và tái tạo bản đồ. Qua đó thấy được tiềm năng của hệ thống hai camera đồng bộ kết hợp với thiết bị đo lường quán tính. Hệ thống sử dụng một camera hoạt động chính (25 frame/s) theo hướng tiếp cận của cách giải quyết bằng camera đơn (Monocular) nhưng có kết hợp với camera phụ chạy với tần số thấp hơn (1 frame/s). Hệ thống tận dụng lợi thế của một camera khi có tầm xử lý xa hơn và tính toán nhanh hơn, với hệ thống hai camera đồng bộ là việc có dữ liệu độ sâu khôi phục dễ dàng và tỷ lệ khoảng cách đã biết theo hệ mét. Trong khi đó vượt qua những khuyết điểm của từng cách tiếp cận, với một camera là không biết tỷ lệ khoảng cách chính xác và dễ bị trôi (drift), với camera đồng bộ là hạn chế về baseline (khoảng cách giữa hai tâm camera) bé và tầm xử lý gần hơn.

Ngoài ra để có thể điều khiển robot, cần có luồng input vị trí của robot với tần số cao mà tính toán với tần số camera (25 frame/s) thì sẽ không thể đáp ứng kịp. Nhận ra được ưu điểm của thiết bị đo lường quán tính (IMU) khi cho ra đầu ra ở tần số cao (100 output/s), hệ thống sử dụng bộ lọc Kalman để kết hợp kết quả tính toán từ ảnh và IMU nâng tần số output vị trí robot lên 100 (output/s). Đồng thời với khả năng xác định góc xoay chính xác, ổn định theo thời gian, hệ thống chia việc xác định chuyển động thành ước lượng góc xoay và phép dịch chuyển, IMU sẽ giúp robot xác định góc xoay chính xác trong khi đó dữ liệu ảnh sẽ chỉ xác định phép dịch chuyển vì vậy giúp mô hình ít phức tạp hơn và có khả năng tính toán nhanh, trong thời gian thực.



Hình 2-1 Mô hình xử lý của hệ thống định vị và tái tạo bản đồ bằng hai camera đồng bộ kết hợp thiết bị đo lường quán tính (IMU).



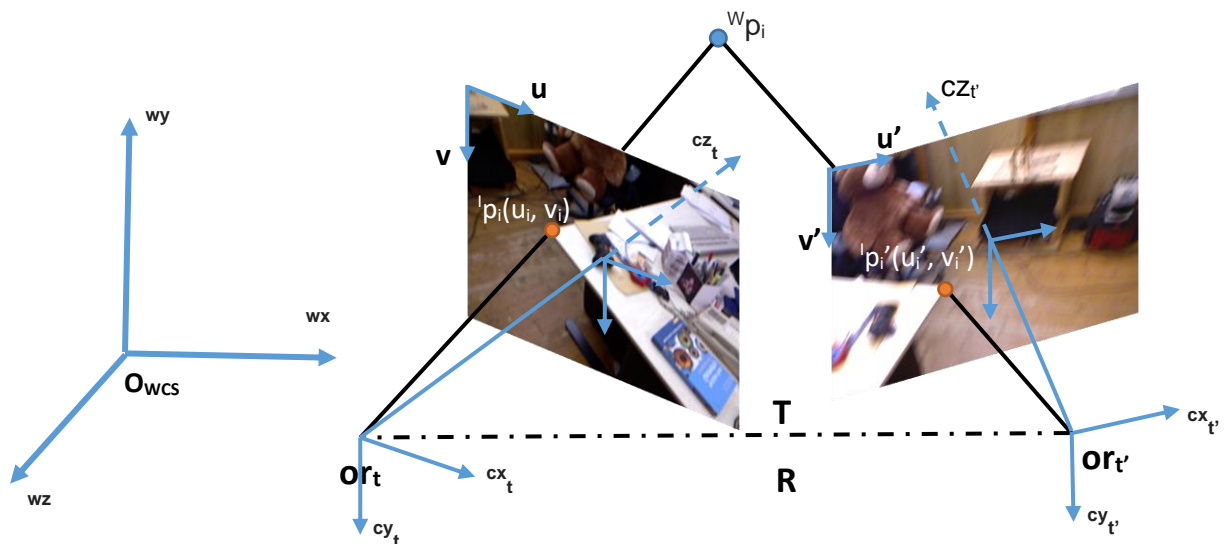
## CHƯƠNG 3: ƯỚC LƯỢNG VỊ TRÍ VÀ TÁI TẠO BẢN ĐỒ LÂN CẬN

*Ở chương này khóa luận sẽ trình bày về cách xử lý dữ liệu từ hai camera và thiết bị đo lường quán tính (Inertial Measurement Unit – IMU) để khôi phục lại vị trí của camera trong không gian 3D (Localization) và bản đồ đặc trưng 3D lân cận (Mapping). Ở đây khóa luận sử dụng IMU để tính toán phép xoay, hình ảnh từ cameras để ước lượng phép dịch chuyển, kết hợp lại để ước lượng vị trí của Robot đồng thời tái tạo bản đồ đặc trưng lân cận (Local map) theo từng thời điểm.*

### **3. Ước lượng vị trí và tái tạo bản đồ lân cận:**

Để có vị trí hiện tại, robot cần biết được nó đã di chuyển theo hướng nào, với khoảng cách là bao nhiêu với thời điểm trước đó. Thông tin này có thể thu được từ thiết bị đo lường quán tính (IMU). Tuy nhiên, thông tin này không chính xác theo thời gian, rất dễ bị trôi (drift) khiến cho ước lượng vị trí robot sai. Bài toán đặt ra là làm sao để ước lượng chuyển động của robot kết hợp thông tin hình ảnh (chính xác hơn) và thông tin từ IMU (nhẹ hơn) nhằm khiến hệ thống hoạt động chính xác và hiệu quả trong thời gian thực.

Về cơ bản, ý tưởng chính của quá trình ước lượng chuyển động bằng thông tin thị giác kết hợp với IMU có thể tóm tắt như sau. Chuyển động đến  $r_t$  sẽ được ký hiệu là  $W_t$  bao gồm một phép tịnh tiến  $T$  và một phép xoay  $R$ . Tại hai vị trí  $r_t$  và  $r_{t'}$  robot cùng quan sát được cùng một điểm trong môi trường thực tế (gọi là điểm tương ứng), tuy nhiên do sự khác nhau về vị trí quan sát và phép chiếu mà ta chỉ biết được



Hình 3-1 Minh họa cho bài toán ước lượng chuyển động bằng thông tin thị giác. Điểm màu xanh là điểm trong không gian ba chiều trên thực tế. Điểm màu đỏ là tọa độ trên ảnh của điểm thực tế khi camera quan sát được tại hai thời điểm khác nhau  $t$ ,  $t'$ .

điểm đó có vị trí  $(u, v)$  trong ảnh tại vị trí  $r_t$  và vị trí  $(u', v')$  trong ảnh tại vị trí  $r_{t'}$ , hai vị trí quan sát cùng một điểm không thay đổi trong không gian ba chiều  ${}^w p(x, y, z)$ . Nhiệm vụ của thuật toán ước lượng chuyển động là tìm một tập điểm tương ứng và vị trí ba chiều của tập điểm, sau đó ước lượng vị trí (phép tịnh tiến  $T$ ) và hướng (phép xoay  $R$ ) thể hiện chuyển động đến  $r_t$  của robot theo hệ tọa độ thế giới (WCS). Ở đây, khóa luận xin trình bày phương pháp ước lượng dựa vào thông tin hình ảnh từ camera kết hợp với IMU. Với sự xuất hiện của dữ liệu IMU, ta có thể biết được chính xác phép quay  $R$  trong phép biến đổi  $W_t$ , và với thông tin hình ảnh từ hai camera để ước lượng phép dịch chuyển  $T$  khi đã biết phép xoay  $R$  từ IMU.

Thông tin hình ảnh đến từ hai camera chính (25Hz) và camera phụ (1Hz) có tần số khác nhau nên thông tin hình ảnh sẽ được chia thành hai hệ thống: camera đơn

(camera chính 25Hz) (monocular) và hệ thống camera đồng bộ (hai camera 1Hz) (stereo). Vì hệ thống chủ yếu sử dụng đặc trưng để tính toán, hai hệ thống trên sẽ có cách tìm đặc trưng khác nhau. Với hệ thống camera đồng bộ là so khớp hai ảnh cùng thời điểm camera chính và phụ để tìm cặp đặc trưng tương ứng, hệ thống camera đơn là theo vết các đặc trưng theo từng thời điểm ở camera chính.

Hệ thống định vị và tái tạo bản đồ (Simultaneous Localization and Mapping - SLAM) gồm hai phần chính là định vị (Localization) và tái tạo bản đồ (Mapping) được thực hiện đồng thời tại mỗi thời điểm  $t$ . Tại thời điểm  $t$ , việc định vị được thực hiện trước dựa trên việc tạo bản đồ thời điểm  $t-1$  trước đó. Sau khi có vị trí Robot, việc tái tạo bản đồ cho thời điểm hiện tại  $t$  được thực hiện. Vì việc định vị được thực hiện trước, nên sẽ có một pha dùng để tạo bản đồ khởi động cho hệ thống định vị và tái tạo bản đồ. Hệ thống gồm các bước chính: Phát hiện và theo vết đặc trưng, khởi động bản đồ, định vị vị trí, cập nhật bản đồ lân cận, phát hiện và phục hồi lỗi, kết hợp định vị bằng IMU.

### 3.1 Phát hiện và theo vết đặc trưng:

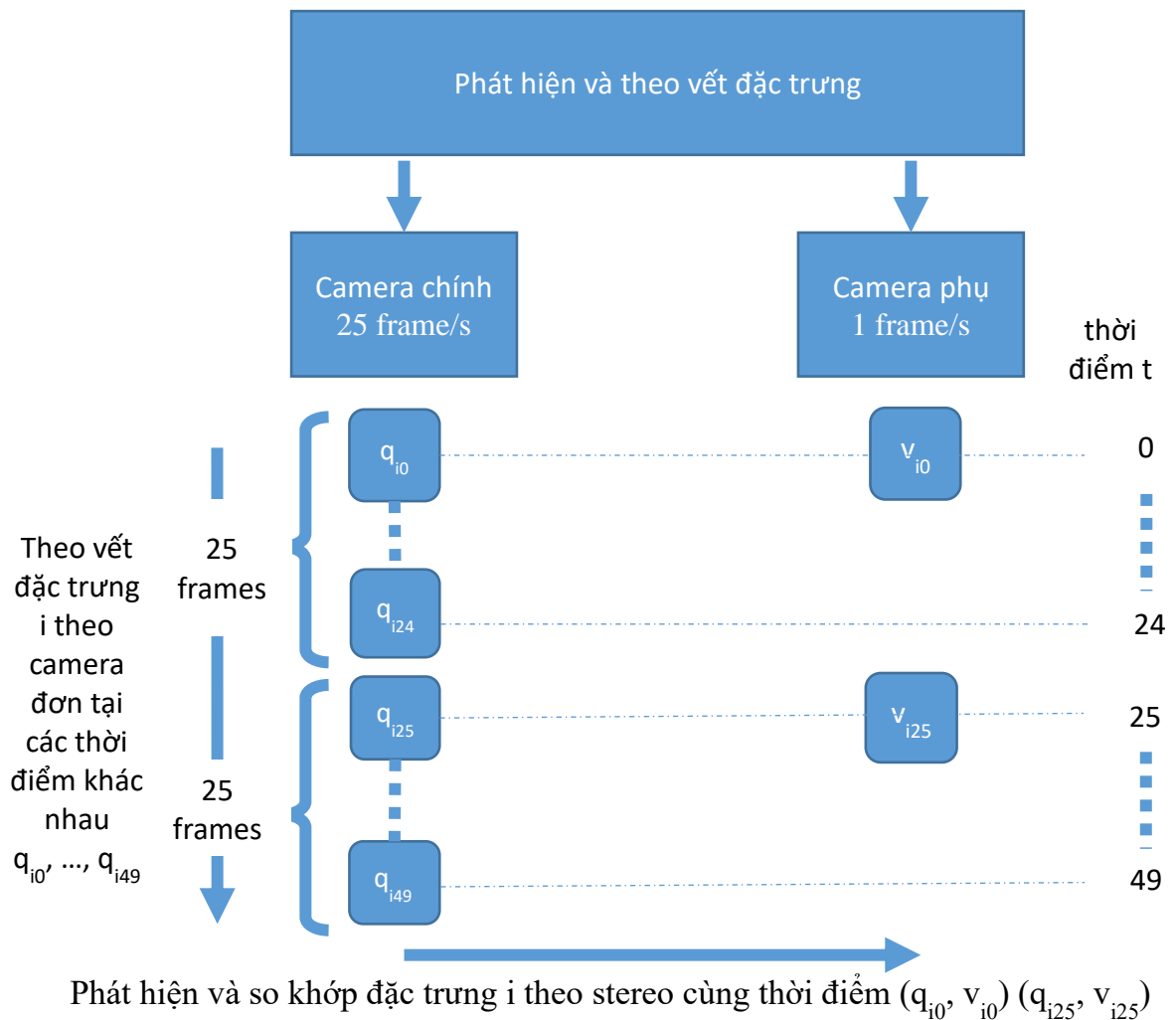
Hệ thống chủ yếu dùng các đặc trưng quan sát được tại mặt phẳng ảnh để xác định vị trí robot, nên việc phát hiện và theo vết đặc trưng là rất quan trọng. Hệ thống gồm hai hệ thống là hệ thống camera đơn (Monocular) và hệ thống camera đồng bộ (Stereo), mỗi hệ thống đều cần đến các đặc trưng để tính toán:

- Hệ thống camera đồng bộ (Stereo) sử dụng những cặp đặc trưng tương đồng  $\mathbf{q}_{it}$  và  $\mathbf{v}_{it}$  từ hai ảnh cùng thời điểm  $t$  (camera chính và phụ) có tác dụng tái tạo vị trí  ${}^w\mathbf{p}_i$  của đặc trưng trong không gian 3D theo hệ tọa độ thế giới WCS. Những điểm được tạo nên này có tác dụng làm điểm khởi tạo cho hệ thống camera đơn (Monocular) và dùng để phát hiện lỗi trong hệ thống.
- Hệ thống camera đơn (Monocular) sử dụng những đặc trưng theo vết theo thời gian của mặt phẳng ảnh camera chính  $\mathbf{q}_{it}$  kết hợp các vị trí 3D (khởi tạo từ hệ thống camera đồng bộ (Stereo)) của các đặc trưng quan sát được trong mặt

phẳng ảnh  $\mathbf{p}_i$  để tính toán vị trí camera  $\mathbf{r}_t$ . Đồng thời khôi phục vị trí 3D của các đặc trưng mà không khôi phục được bằng hệ thống camera đồng bộ.

Quá trình phát hiện và theo vết đặc trưng được thực hiện như sau: hệ thống gồm hai camera, một camera chính chạy với tần số 25Hz và một camera phụ với tần số 1Hz. Việc phát hiện đặc trưng chỉ được thực hiện ở hệ thống camera đồng bộ (stereo) với tần số 1Hz, tức là chỉ khi có ảnh từ camera chính và phụ, việc phát hiện đặc trưng mới được thực hiện. Với ảnh từ camera chính phát hiện ra tập đặc trưng  $q_{it}$  và với ảnh từ camera phụ sẽ là tập đặc trưng  $v_{it}$  (đặc trưng thứ  $i$ , thời điểm  $t$ ). Sau đó, giữa hai lần phát hiện đặc trưng, chỉ có ảnh đến từ camera chính, hệ thống camera đơn (monocular) sẽ theo vết đặc trưng  $q_{it}$  trong ảnh camera chính tiếp theo là  $q_{i, t+1}$ . Việc này nhằm tiết kiệm độ phức tạp tính toán cho hệ thống khi việc phát hiện đặc trưng chỉ được thực hiện 1Hz và việc theo vết đặc trưng gần như không tốn chi phí khi độ dịch chuyển giữa các khung hình là rất bé.

### 3.1.1 Hệ thống camera đồng bộ (stereo):



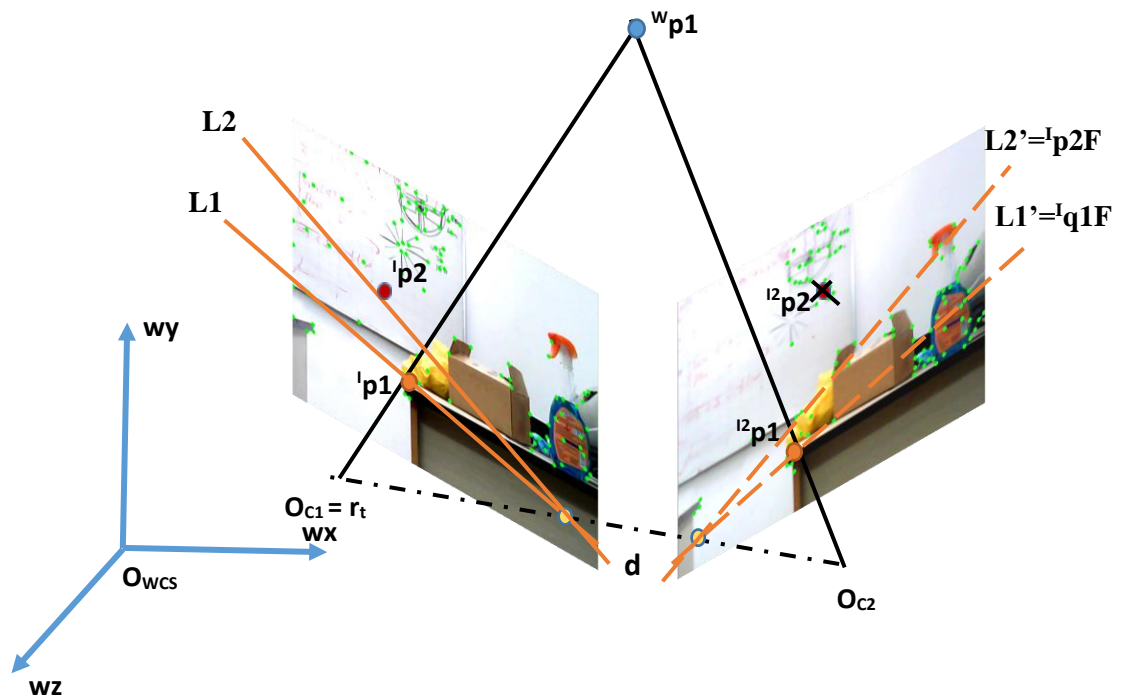
Hình 3-2 Phát hiện và theo vết đặc trưng của hai camera từng thời điểm

#### 3.1.1.1 Phát hiện đặc trưng:

Các đặc trưng thường được sử dụng nhất như SIFT, SURF, ... có khả năng bất biến với một số phép biến đổi affine như phép xoay, phép co giãn, ... nhưng có chi phí tính toán rất lớn nên không thích hợp bài toán đặt ra. Ở đây, khóa luận sử dụng đặc trưng góc Shi-Tomasi phát hiện các góc trong ảnh hai chiều bằng việc tính được thay đổi độ xám theo các hướng, đặc trưng này có độ phức tạp tính toán thấp hơn với SIFT, SURF, ... nhưng vẫn đảm bảo tính bất biến với một số phép affine cũng như đặc tính dễ dàng theo vết về sau. Việc bất biến với phép affine rất có lợi với bài

toán Visual SLAM khi camera có thể quay hoặc chuyển động tịnh tiến và vẫn có thể đối sánh được các đặc trưng.

Sau khi phát hiện đặc trưng góc Shi-Tomashi, cần một miêu tả tốt cho đặc trưng để có thể tìm các cặp tương ứng giữa hai ảnh từ hai camera. Đặc trưng được mô tả bằng BRIEF description, ý tưởng chính của BRIEF là mô tả vùng xung quanh đặc trưng bằng chuỗi các bit, việc mô tả này khiến BRIEF có thời gian tính toán rất nhanh cũng như khiến việc so khớp sau giảm phức tạp. BRIEF cũng bất biến với một số phép biến đổi affine.

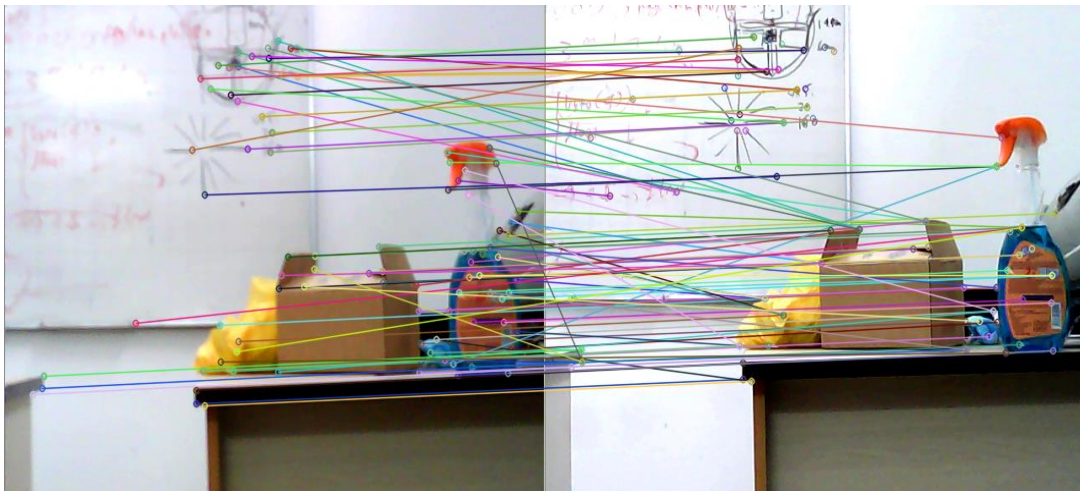


Hình 3-3 Các điểm màu xanh là đặc trưng góc Shi-Tomasi tại camera chính và phụ cùng thời điểm  $t$ . Điều kiện ràng buộc để thỏa mãn cặp tương đồng epipolar là một điểm phải nằm trên đường thẳng epipolar line tương ứng với điểm tương đồng.  $p_1^2$  nằm trên đường thẳng epipolar  $L_2'$  (màu cam) tương ứng với  $p_1^1$  nên  $p_1^1$  và  $p_1^2$  thỏa mãn điều kiện cặp tương đồng. Trong khi đó  $p_2^2$  không nằm trên  $L_2'$  là epipolar tương ứng của  $p_2^1$  nên không phải là cặp tương đồng.

### 3.1.1.2 Đối sánh đặc trưng:

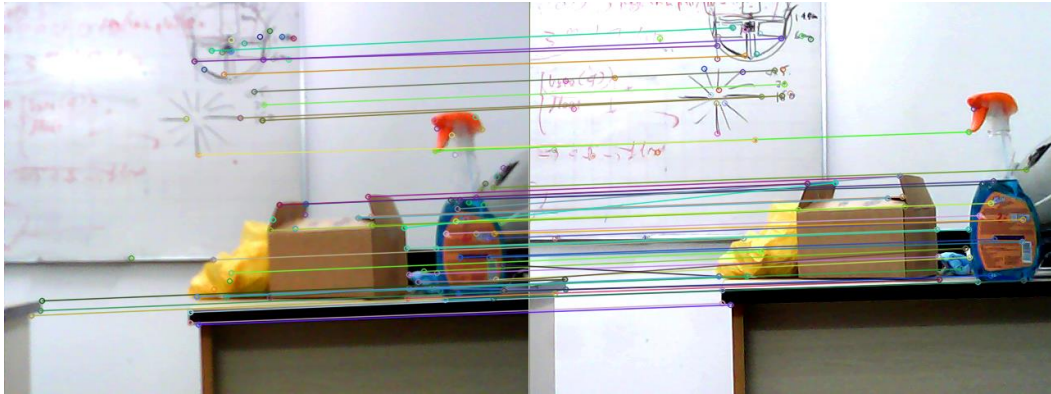
Sau khi đã rút trích đặc trưng từ hai ảnh, việc tiếp theo là đối sánh đặc trưng tìm các cặp tương đồng. Bước so khớp này được thực hiện thông qua tìm kiếm láng giềng gần nhất dựa trên khoảng cách Hamming của các vector đặc trưng. Với mỗi đặc trưng trong ảnh nguồn, ta sẽ tìm được đặc trưng có khoảng cách gần nhất trong ảnh gốc.

Do hai đặc trưng nếu là quan sát của cùng một điểm 3D lên mặt phẳng ảnh thì thỏa mãn giới hạn Epipolar:  $(l^1p1) F(l^2p1) = 0$ . Với  $F$  là fundamental matrix,  $l^1p1$  và  $l^2p1$  tương ứng vị trí pixel của đặc trưng tại ảnh từ hai camera, với ý nghĩa  $(l^1p1) F$  là epipolar line  $L1'$  trên mặt phẳng ảnh camera phụ mà  $l^2p1$  sẽ nằm trên đường thẳng này, nếu  $l^2p1$  không thuộc  $L1'$  tích  $(l^1p1) F(l^2p1)$  sẽ lớn và  $l^1p1$ ,  $l^2p1$  được xác định là nhiễu. Việc loại bỏ các điểm nhiễu (outlier) có thể được thực hiện bằng so sánh tích trên với một ngưỡng, ở đây là 0.1, nếu lớn hơn ngưỡng thì sẽ xác định là outlier và loại cặp điểm đó ra. Ý nghĩa của việc này là nhằm loại bỏ những cặp điểm bị nhầm lẫn trong quá trình so khớp do sai số bởi tính đồng nghĩa và đa nghĩa của đặc trưng thị giác.



Hình 3-4 So khớp giữa hai ảnh từ camera chính và phụ cùng thời điểm. Bên trái là ảnh từ camera chính, bên phải là ảnh camera phụ. Các đường nối vị trí của các cặp tương ứng.





Hình 3-5 Các cặp đặc trưng sau khi đã loại nhiễu trong việc so khớp

### 3.1.2 Hệ thống camera đơn (monocular):

#### 3.1.2.1 Theo vết đặc trưng từ camera chính:

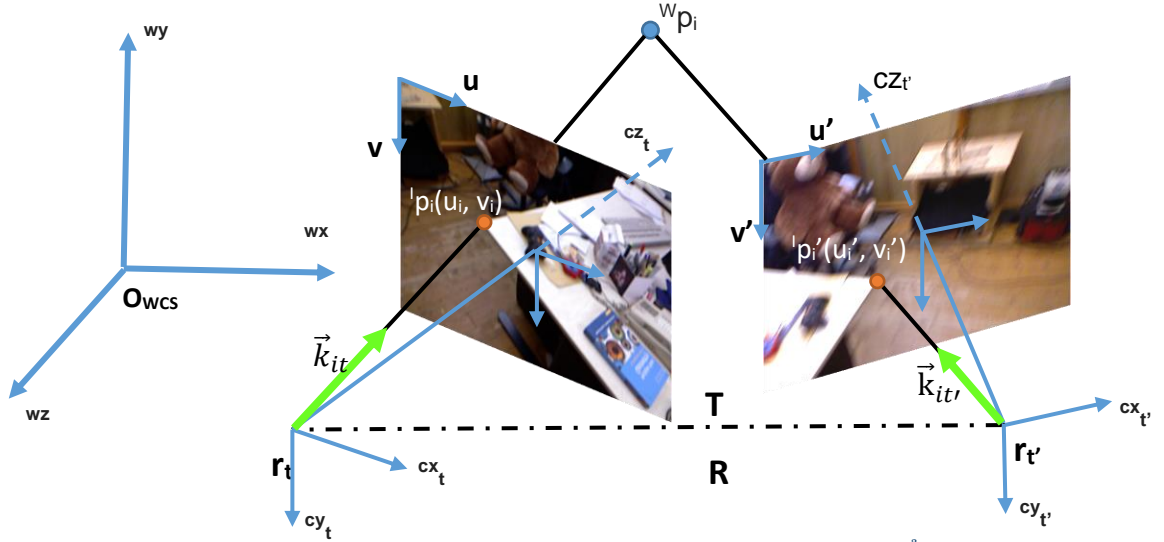


Hình 3-6 Theo vết đặc trưng theo từng khung hình đến từ

Với dãy ảnh từ camera chính (25Hz), để có vị trí của đặc trưng theo từng thời điểm  $q_{it}$  nếu thực hiện việc phát hiện đặc trưng và đối sánh với hai ảnh từ hai thời điểm liên tiếp. Với dãy ảnh từ camera chính (25Hz), để có vị trí của đặc trưng theo từng thời điểm  $q_{it}$  nếu thực hiện việc phát hiện đặc trưng và đối sánh với hai ảnh từ hai thời điểm liên tiếp thì sẽ tốn rất nhiều chi phí tính toán do phải lặp thực hiện các bước phát hiện, rút trích, đối sánh đặc trưng, ... Vì thế luận văn ở đây sử dụng phương pháp theo vết Kanade-Lucas-Tomasi Feature Tracker (KLT) để theo vết đặc trưng qua từng khung ảnh. Phương pháp dựa trên tìm kiếm một vùng (patch) xung quanh đặc trưng theo đạo hàm của nó. Do độ dịch chuyển pixel của đặc trưng giữa các khung hình là rất bé, nên KLT tốn rất ít chi phí.



### 3.1.2.2 Vector quan sát của đặc trưng:



Hình 3-7 Minh họa cho quan sát của đặc trưng  ${}^w p_i$  tại hai thời điểm  $t, t'$  trên mặt phẳng ảnh camera chính. Vector màu xanh  $\vec{k}_{it}$  và  $\vec{k}_{it'}$  lần lượt là quan sát của  $p_i$  tại  $t, t'$ . Các vector trên có hướng từ vị trí robot từng thời điểm ( $r_t = O_{(C1)_t}$ ) chỉ đến vị trí đặc trưng  $p_i$  trong tọa độ thế giới. Vector quan sát của đặc trưng có thể tính từ vị trí đặc trưng trên mặt phẳng ảnh camera chính.

Các quan sát đặc trưng đóng vai trò rất quan trọng với hệ thống, mỗi đặc trưng sẽ cung cấp thông tin về hướng từ vị trí camera đến vị trí đặc trưng thông qua quan sát tại mặt phẳng ảnh từng thời điểm. Thông tin đó được dùng tìm lại vị trí camera. Cách tính được vector quan sát (unit feature observation) gồm các bước:

- 1- Quan sát của đặc trưng  $i$  từ hệ tọa độ mặt phẳng ảnh thuần nhất  ${}^I p_i(u_i, v_i, 1)$  sẽ được chuyển thành vector quan sát ở hệ tọa độ Camera (C)  $\vec{k}_{it}^c$ :

$$\vec{k}_{it}^c = K^{-1} {}^I p_i$$

Sau đó đưa về dạng vector đơn vị:

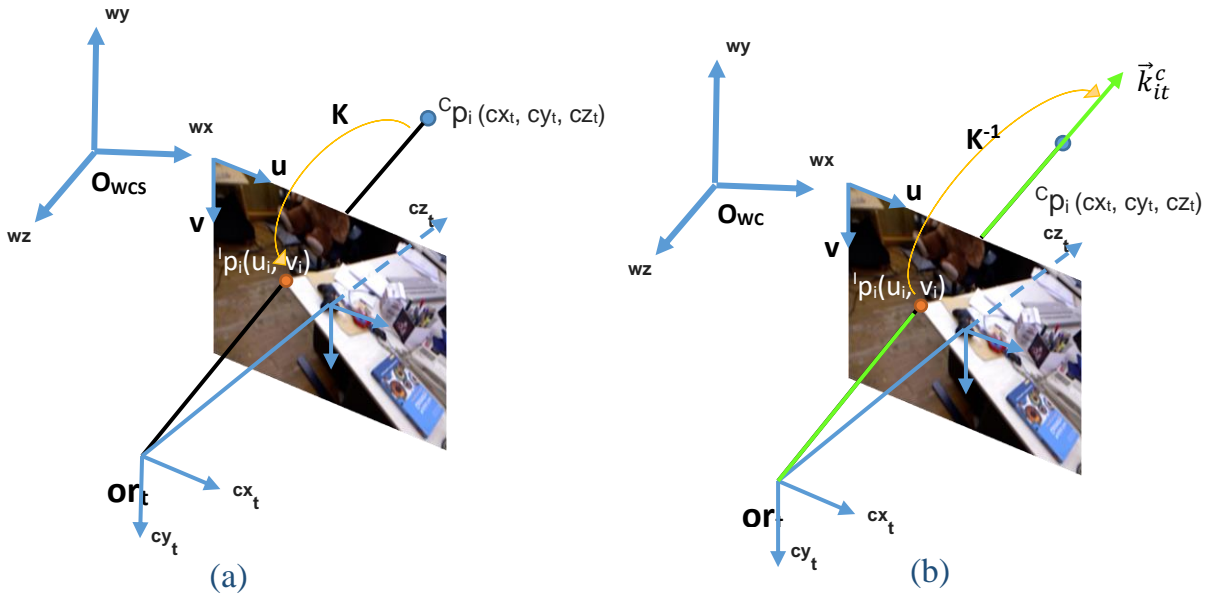
$$\vec{k}_{it}^c = \frac{\vec{k}_{it}^c}{\|\vec{k}_{it}^c\|} \quad (3.1)$$

Với  $K$  là phép biến đổi từ hệ tọa độ camera (C) vào hệ tọa độ mặt phẳng ảnh:

$$K = \begin{pmatrix} f_u & 0 & u_c \\ 0 & f_v & v_c \\ 0 & 0 & 1 \end{pmatrix} \quad \text{với: } (u_c, v_c): \text{ tâm chiếu}$$

$f_u = f_v = f$ : tiêu cự theo chiều  $u, v$

$K$  là ma trận camera chiếu điểm  ${}^C p_i (c_{x_t}, c_{y_t}, c_{z_t})$  thuộc tọa độ Camera xuống tọa độ mặt phẳng ảnh là  ${}^I p_i (u_i, v_i)$ . Ma trận nghịch đảo của  $K$  là  $K^{-1}$  sẽ chiếu một điểm  ${}^I p_i (u_i, v_i)$  trên mặt phẳng ảnh thành vector chỉ hướng đến  ${}^C p_i$  trong hệ tọa độ camera (C). Sau đó vector chỉ hướng sẽ được chuyển thành vector đơn vị  $\vec{k}_{it}^c$ .



Hình 3-8 a) Minh họa cho phép chiếu  $K$  chiếu một điểm từ hệ tọa độ camera (C)  ${}^C p_i$  xuống mặt phẳng ảnh.

b) Minh họa cho phép chiếu  $K^{-1}$  chiếu một điểm trên mặt phẳng ảnh thành vector có hướng từ tâm camera  $O_{(C) t}$  qua điểm 3D của nó ở tọa độ camera  ${}^C p_i$ .

2- Chuyển hệ trục cho vector hướng  $\vec{k}_{it}^c$  từ hệ tọa độ camera (C) sang tọa độ thế giới (W)  $\vec{k}_{it}$ :

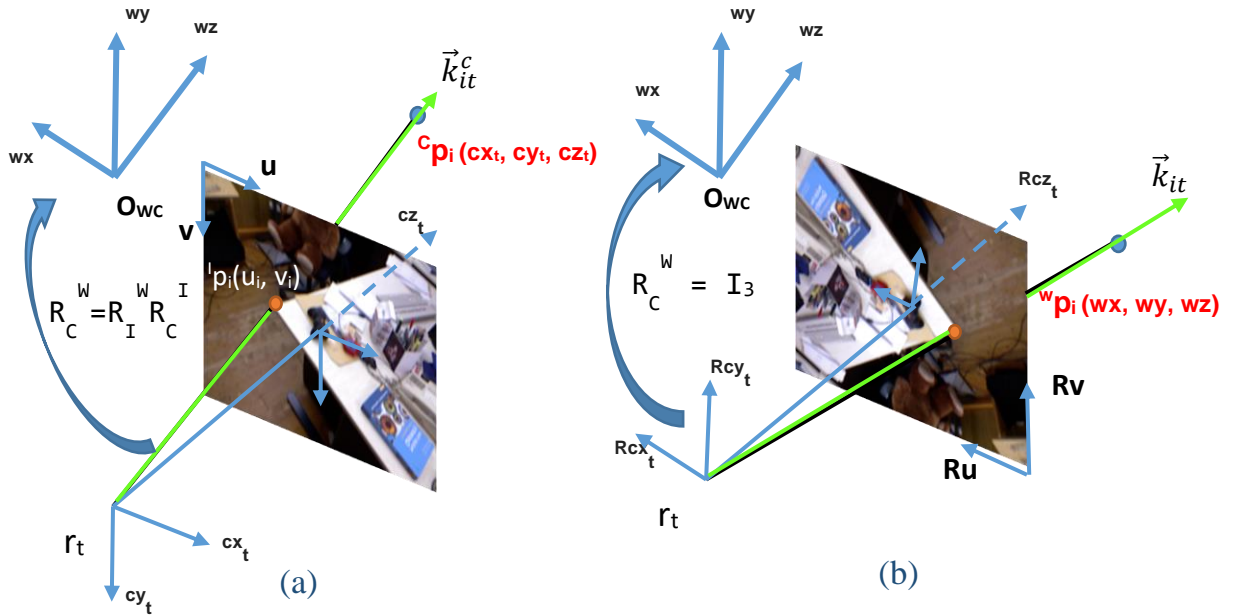
$$\vec{k}_{it} = R_I^W R_C^I \vec{k}_{it}^c \quad (3.2)$$

với:

- $R_C^I$ : phép xoay từ camera  $\rightarrow$  IMU (có được khi cân chỉnh camera)
- $R_I^W$ : phép xoay từ IMU  $\rightarrow$  world (W) (có được từ IMU tại mỗi thời điểm)

Ở đây, ta quy ước phép xoay  $R_C^I = I_3$ , tức là hệ trục camera đối với hệ trục IMU không chứa phép xoay. Nên công thức trên có thể viết thành:  $\vec{k}_{it}^c = R_I^W \vec{k}_{it}^c$ .

Vì  $\vec{k}_{it}^c$  là vector hướng, nên việc chuyển sang hệ tọa độ thế giới chỉ cần phép xoay giữa hai hệ tọa độ. Nếu thêm phép dịch chuyển vào cũng sẽ cho ra vector có hướng cùng với việc chỉ thực hiện phép xoay. Ý nghĩa chuyển  $\vec{k}_{it}^c$  sang hệ tọa độ thế giới: điều chỉnh hướng của vector từ tâm camera đến điểm 3D ở tọa độ camera  $c_{pi}$  sang từ tâm camera đến điểm 3D ở tọa độ thế giới  $w_{pi}$ . Lý do của việc này là vì cần tính tọa độ 3D của đặc trưng tại một hệ tọa độ chung là WCS.



Hình 3-9 a) Minh họa phép xoay từ hệ trục camera sang hệ trục thế giới.

b) Khi áp dụng phép xoay vào, vector quan sát  $\vec{k}_{it}^c$  trở thành  $\vec{k}_{it}$  và có hướng từ tâm camera qua vị trí 3D của đặc trưng thứ  $i$  trong hệ tọa độ thế giới

### 3.2 Khởi động bản đồ lân cận (local map):

Như đã nói ở trên, để ước lượng vị trí camera thời điểm hiện tại (Localization) phải có bản đồ lân cận (Mapping) thời điểm trước đó, và sau khi tính vị trí camera (Localization) lại thực hiện cập nhật bản đồ lân cận thời điểm hiện tại (Mapping). Vì vậy phải có một bản đồ khởi động để bắt đầu quy trình xác định vị trí và tái tạo bản

đồ (SLAM). Tại thời điểm thứ  $t = 0$ , có hai ảnh từ hai camera chính và phụ, ta có thể tính được thông tin vị trí 3D của đặc trưng với hệ thống camera đồng bộ (stereo) và dùng những điểm 3D này khởi tạo bản đồ lân cận.

Khi khởi động, robot chưa biết gì về thông tin môi trường, ngoài các quan sát đặc trưng đến từ chuỗi ảnh camera chính và phụ. Việc tái tạo thông tin ba chiều từ quan sát của hai camera đầu tiên đóng vai trò khởi động hệ thống bắt đầu biết khoảng cách metric và tính được vị trí robot. Các lần tiếp theo tái tạo thông tin ba chiều từ stereo đóng vai trò kiểm tra lỗi hệ thống và phục hồi khi có lỗi.

Sau bước đối sánh đặc trưng, hệ thống đã có được tập đặc trưng  $q_{it}$  ở ảnh camera chính và tập tương ứng  $v_{it}$  ảnh từ camera phụ. Việc tính toán vị trí 3D được thực hiện theo Linear Triangulation Methods, đây là một cách tính đơn giản được Hartley và Zisserman đề xuất.

Trong quá trình căn chỉnh camera, ta đã biết được phép chiếu từ không gian 3D theo hệ tọa độ camera chính xuống ảnh ở hai camera chính và phụ là  $P$  và  $P'$ :  $q_{it} = P \cdot c_{pit}$ ,  $v_{it} = P' \cdot c_{pit}$  với  $c_{pit}$  là tọa độ 3D theo hệ trục camera chính,  $q_{it}$ ,  $v_{it}$  là hình chiếu xuống mặt phẳng ảnh camera chính và phụ của  $X$ . Từ hai phương trình trên, có thể viết lại thành dạng tuyến tính theo  $X$ :  $AX = 0$  và giải theo phương pháp cực tiểu bình phương (least square) SVD để tìm  $X$ .

$$A \cdot c_{pit} = \begin{bmatrix} uP^{3T} - P^{1T} \\ vP^{3T} - P^{2T} \\ u'P'^{3T} - P'^{1T} \\ v'P'^{3T} - P'^{2T} \end{bmatrix} c_{pit} = 0 \quad (3.3)$$

Sau khi tìm được vị trí 3D  $c_{pit}$  của các cặp đặc trưng tương ứng  $q_{it}$ ,  $v_{it}$  với hệ thống camera đồng bộ (stereo), ta sẽ chuyển sang hệ trục tọa độ thế giới WCS.

Vì định nghĩa vị trí camera chính thời điểm thứ  $t = 0$  cũng là vị trí robot thời điểm  $t = 0$  là  $r_0 = (0, 0, 0)$ . Nên ta chỉ cần áp dụng phép xoay có được từ IMU để chuyển hệ tọa độ các điểm trên  $c_{pit}$  từ hệ trục tọa độ camera chính (C) sang hệ trục tọa độ thế giới WCS:

$${}^w p_i = R_I^W R_C^I c_{pit} \quad (3.4)$$

Như đã nói ở trên,  $R_C^I = I_3$  nên ta chỉ cần áp dụng phép xoay từ IMU sang hệ tọa độ thế giới có được ở mỗi thời điểm. Các vị trí này sẽ được thêm vào bản đồ lân cận cùng với quan sát của nó như là khởi tạo.

### 3.3 Xác định vị trí robot:

Tại thời điểm hiện tại  $t$ , hệ thống đã có bản đồ lân cận từ thời điểm trước đó  $t-1$ , hệ thống sẽ tính toán vị trí robot dựa trên vị trí 3D của đặc trưng và quan sát của đặc trưng tại thời điểm hiện tại  $t$ .

Vị trí của robot cũng chính là vị trí camera chính (như quy ước mục 3.1.1), ký hiệu  $r_t$  sẽ được tính bằng cực tiểu hóa độ lỗi bình phương phép reprojection của tất cả feature trong bản đồ 3D đặc trưng lân cận:

$$r_t = \underset{r_t}{\operatorname{argmin}} \sum_{i \in \mathfrak{S}} \left\| \frac{r_t - {}^w p_i}{\|r_t - {}^w p_i\|} \times \vec{k}_{it} \right\|^2 \quad (3.5)$$

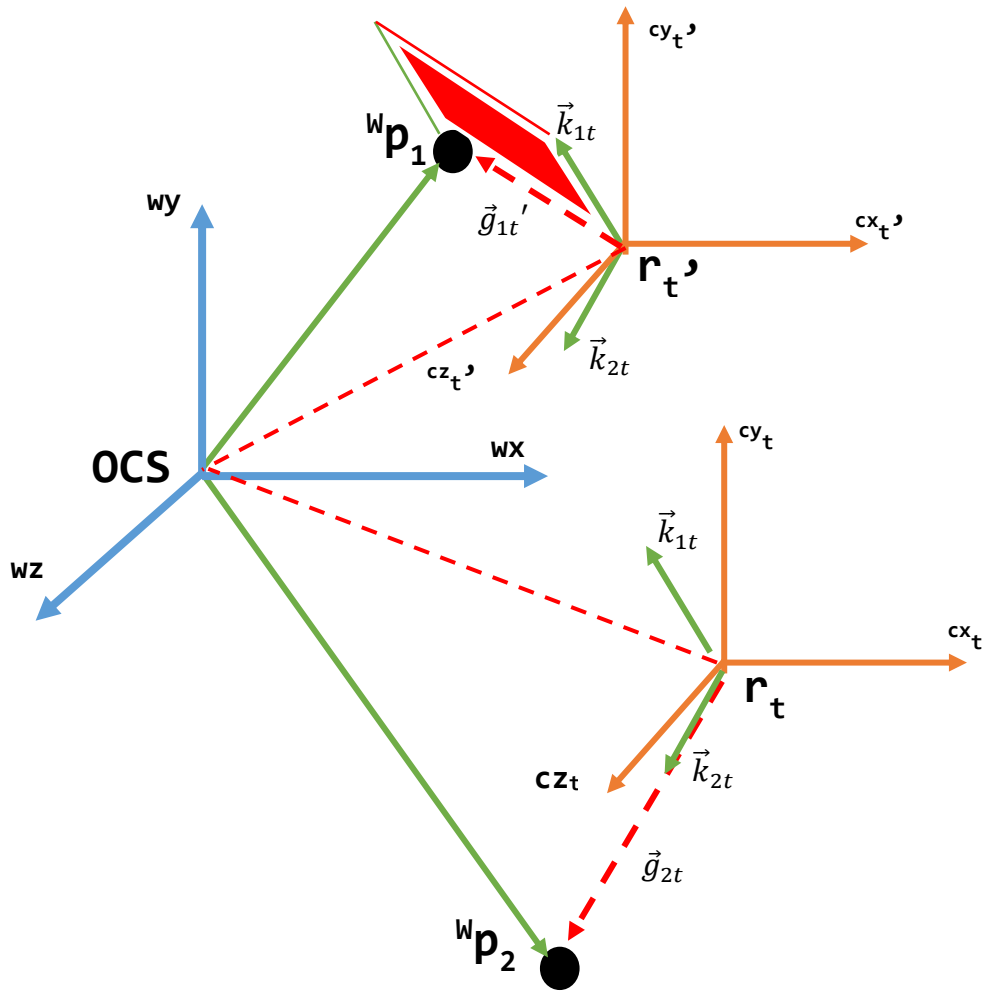
Diễn giải:

- $\vec{k}_{it}$  là vector quan sát của đặc trưng thứ  $i$  thời điểm  $t$ , cũng vector nối từ đặc trưng  $i$  đến  $r_t$ .
- $\frac{r_t - {}^w p_i}{\|r_t - {}^w p_i\|}$  ký hiệu  $\vec{g}_{it}$  là vector đơn vị thật sự khi có vị trí chính xác của  $r_t$ .
- $\mathfrak{S}$  là tập các đặc trưng quan sát được tại ảnh camera chính thời điểm  $t$ .

Tích có hướng giữa  $\vec{g}_{it} \times \vec{k}_{it}$  tỷ lệ với diện tích hình bình hành tạo ra từ 2 vector. Do đó, khi góc giữa hai vector là bé thì tích này sẽ bé và ngược lại. Mục đích ở đây là cực tiểu sai số giữa các quan sát ( $\vec{k}_{it}$ ) và vector thật sự ( $\vec{g}_{it}$ ) khi tìm được  $r_t$  bằng cách tìm vị trí  $r_t$  trong không gian để có  $\vec{g}_{it}$  có hướng gần nhất với  $\vec{k}_{it}$  đã biết.


Lưu ý rằng, do độ dịch chuyển camera giữa các khung ảnh là rất bé, nên ta có thể tính xấp xỉ:


$$d_i = \|r_t - {}^w p_i\| \approx \|r_{t-1} - {}^w p_i\| \quad (3.6)$$



Hình 3-10 Minh họa các vị trí camera thời điểm t và sai số so với vector quan sát  $\vec{k}_{it}$ . Vị trí có sai số nhỏ nhất (tổng diện tích hình bình hành đồ nhỏ nhất) sẽ là vị trí robot thời điểm t. Ở đây là vị trí  $r_t$  có sai số nhỏ nhất so với vector quan sát được.

Có  $r_t$  và  $r'_t$  là các điểm trong vô số điểm của camera có thể trong không gian tại thời điểm t, ta có  ${}^w p_1$  và  ${}^w p_2$  là các vị trí 3D của đặc trưng và  $\vec{k}_{1t}, \vec{k}_{2t}$  là các quan sát của đặc trưng tại thời điểm t. Mục đích là tìm  $r_t$  sao cho tất cả vector  $\frac{r_t - {}^w p_i}{\|r_t - {}^w p_i\|} = \vec{g}_{it}$  có hướng gần với vector  $\vec{k}_{it}$  tương ứng nhất.

Khi chọn camera ở vị trí  $r'_t$  (sai) thì ta sẽ tính được vector  $\vec{g}_{1t}'$  (  ) và khi tích vô hướng với  $\vec{k}_{1t}$  sẽ ra giá trị độ lớn tương ứng phần hình bình hành màu đỏ lớn (diện tích càng lớn -> độ lỗi càng nhiều).

Khi chọn camera ở vị trí  $r_t$  (đúng) thì vector  $\vec{g}_{2t}$  (  ) nhân tích vô hướng với  $\vec{k}_{2t}$  sẽ ra giá trị độ lỗi gần bằng 0 (do hướng đúng,  $S_{\text{BìnhHành}} = 0$ ). Do đó sai số tại vị trí  $r_t$  là bé nhất nên vị trí cần tìm sẽ là  $r_t$ .

Từ (3.6) chuyển về dạng tuyến tính tìm  $r_t$ :

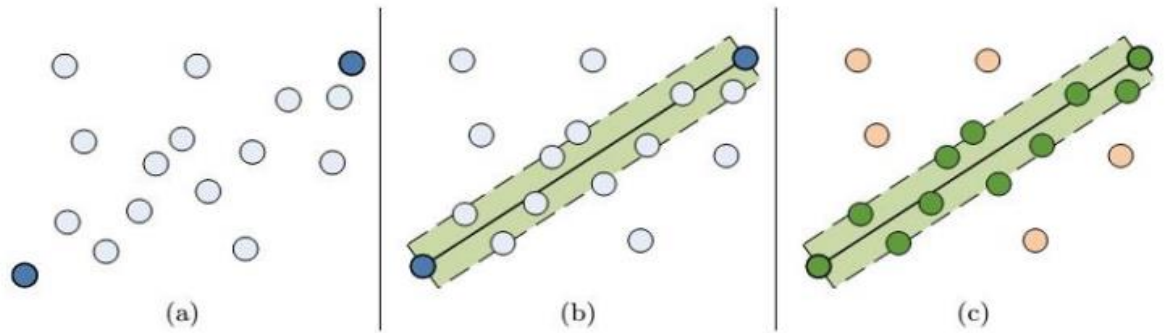
$$\left( \sum_{i \in \mathfrak{S}} \frac{\mathbb{I}_3 - \vec{k}_{it} \vec{k}_{it}^T}{d_i} \right) r_t = \sum_{i \in \mathfrak{S}} \frac{\mathbb{I}_3 - \vec{k}_{it} \vec{k}_{it}^T}{d_i} {}^w p_i \quad (3.7)$$

Với:

- $d_i \approx \|r_{t-1} - {}^w p_i\|$  : giá trị đã biết
- $\mathfrak{S}$ : là tập các đặc trưng quan sát được tại ảnh camera chính thời điểm  $t$ .
- ${}^w p_i$  : vị trí 3D của đặc trưng thứ  $i$  hệ trục tọa độ thế giới WCS.
- $r_t$ : giá trị cần tìm, vị trí robot thời điểm  $t$ .

### 3.3.1 Loại nhiễu sử dụng RANSAC:

Với công thức (3.7), ít nhất có thể dùng hai đặc trưng để tính toán vị trí  $r_t$ . Vì thế có thể sử dụng two-points RANSAC để khử nhiễu (outliers). Đây là một phương pháp rất thường được sử dụng trong lĩnh vực thị giác máy tính để ước lượng một mô hình mà trong đó có tồn tại nhiễu. Ở mỗi lần lặp, một tập mẫu rất nhỏ (vừa đủ để ước lượng phép biến đổi) sẽ được chọn ngẫu nhiên từ các cặp điểm. Sau đó phép biến đổi ước lượng được từ tập mẫu này sẽ được đánh giá dựa trên một hàm độ lỗi. Các bước trên được lặp lại nhiều lần với mỗi lần ta chọn ra một tập mẫu mới và giữ lại phép biến đổi tốt nhất. Sau một số lần lặp, RANSAC sẽ đảm bảo hội tụ một phép biến đổi tốt, nhưng không thể nào chắc chắn đây là phép biến đổi tối ưu. Ưu điểm của RANSAC chính là khả năng ước lượng một mô hình đủ tốt và loại bỏ được nhiễu trong tập dữ liệu. Đây cũng là nguyên nhân mà khóa luận đã lựa chọn phương pháp này để ước lượng chuyển động của camera. Ở đây số lần lặp của RANSAC sẽ được chọn là **50**.



Hình 3-11 Minh họa thuật toán RANSAC để khớp một đường thẳng lên tập dữ liệu hai chiều. (a) Đầu tiên,  $k$  điểm được chọn ngẫu nhiên từ tập dữ liệu ( $k = 2$ ). (b) Từ những điểm này, ta sẽ cho ra được một mô hình. Ở đây, một đường thẳng được vẽ qua hai điểm và các điểm nằm trong vùng màu xanh sẽ được xem là phù hợp với mô hình. (c) Với mỗi điểm trong tập dữ liệu ta sẽ tìm sai số của điểm đó so với mô hình (khoảng cách từ điểm đến đường thẳng). Nếu sai số này là đủ nhỏ thì ta xem các điểm này là phù hợp với mô hình (điểm màu xanh) và loại bỏ các điểm không phù hợp (điểm màu đỏ). Mô hình cuối cùng sẽ được tính lại chỉ dựa vào các điểm màu xanh.

### **Giải thuật 1** Tìm vị trí robot với RANSAC

**Đầu vào:** Bản đồ lân cận  $\mathcal{Q}$ , gồm các thành phần: tập điểm vị trí 3D của các đặc trưng:  ${}^wP = \{{}^w p_i = (w_x, w_y, w_z)^T\}$  và các quan sát tương ứng tại thời điểm hiện tại  $t$ :  $K = \{\vec{k}_{it}\}$ .

**Đầu ra:** Vị trí robot thời điểm hiện tại  $r_t$ .

**Mã giả:**

bestRobotPose, bestInliers  $\leftarrow \emptyset$

**for** iteration = 1 to  $N$  **do**

samples  $\leftarrow$  chọn ngẫu nhiên hai đặc trưng trong bản đồ lân cận  $\mathcal{Q}$

Tính vị trí robot currentRobotPose từ hai điểm samples  $r_t$

inliers  $\leftarrow \emptyset$ , rmse  $\leftarrow 0$

**for** each  ${}^w p_i \in {}^w P$  **do**

$$\text{error} \leftarrow \left\| \frac{r_t - {}^w p_i}{\|r_t - {}^w p_i\|} \times \vec{k}_{it} \right\|^2$$



```

if error < threshold then // quan sát  $u_{ij}$  phù hợp với vị trí  $r_j$ 
    inliers  $\leftarrow$  inliers + ( $^W p_i, \vec{k}_{it}$ )
    rmse  $\leftarrow$  rmse + error
end if
end for
rmse  $\leftarrow$  sqrt(rmse/|inliers|) // căn bậc hai trung bình bình phương sai số
if |inliers| > requiredInliers then
    Tính lại vị trí robot currentRobotPose từ tập inliers:  $r_t$ 
    Tính lại tập inliers từ vị trí robot  $r_t$ 
    if currentRobotPose tốt hơn bestRobotPose then
        bestRobotPose  $\leftarrow$  currentRobotPose
        bestInliers  $\leftarrow$  inliers
    end if
end if
end for
return bestRobotPose, bestInliers

```

Để tìm vị trí robot thời điểm hiện tại  $t$ , ta sử dụng công thức tuyến tính (3.7) tính toán với tập  $\mathfrak{S}$  gồm các đặc trưng dùng để tính:

$$\left( \sum_{i \in \mathfrak{S}} \frac{\mathbb{I}_3 - \vec{k}_{it} \vec{k}_{it}^T}{d_i} \right) r_t = \sum_{i \in \mathfrak{S}} \frac{\mathbb{I}_3 - \vec{k}_{it} \vec{k}_{it}^T}{d_i} {}^W p_i \quad (3.8)$$

Để đánh giá chất lượng của vị trí vừa ước lượng được, ta sẽ dựa vào hai tiêu chí sau:

- + Số lượng các đặc trưng thỏa vị trí robot hiện tại (càng lớn càng tốt)
- + Căn bậc hai trung bình bình phương sai số (*root mean squared error*) khi lấy độ lớn tích vô hướng của vector quan sát  $\vec{k}_{it}$  và vector thực sự  $\vec{g}_{it}$  sử dụng vị trí robot này (càng nhỏ càng tốt).

Việc sử dụng cả hai tiêu chí thay vì chỉ xét đến sai số giữa hai tập điểm là cần thiết. Vì quá trình tính sai số chỉ xét đến các điểm thỏa phép biến đổi nên sai số ít

không chắc chắn rằng đây sẽ là một phép biến đổi tốt. Một phép biến đổi được xem là đủ tốt nếu số lượng cặp điểm thỏa phép biến đổi lớn hơn **20** và trung bình sai số ít hơn **5 cm**. Việc so sánh giữa hai phép biến đổi sẽ dựa vào cả hai tiêu chí này. Phép biến đổi nào tốt hơn ở cả hai tiêu chí thì mới được xem là tốt hơn.

### 3.4 Cập nhật bản đồ lân cận (Mapping):

Bản đồ lân cận được định nghĩa là bản đồ vị trí 3D của các đặc trưng vẫn còn quan sát được trên ảnh camera chính, tức là các đặc trưng chưa bị mất dấu trong quá trình theo vết. Cập nhật bản đồ lân cận có các nội dung: thêm đặc trưng, xóa đặc trưng.

#### 3.4.1 Thêm đặc trưng vào bản đồ lân cận:

+ Với hệ thống camera đồng bộ (stereo):

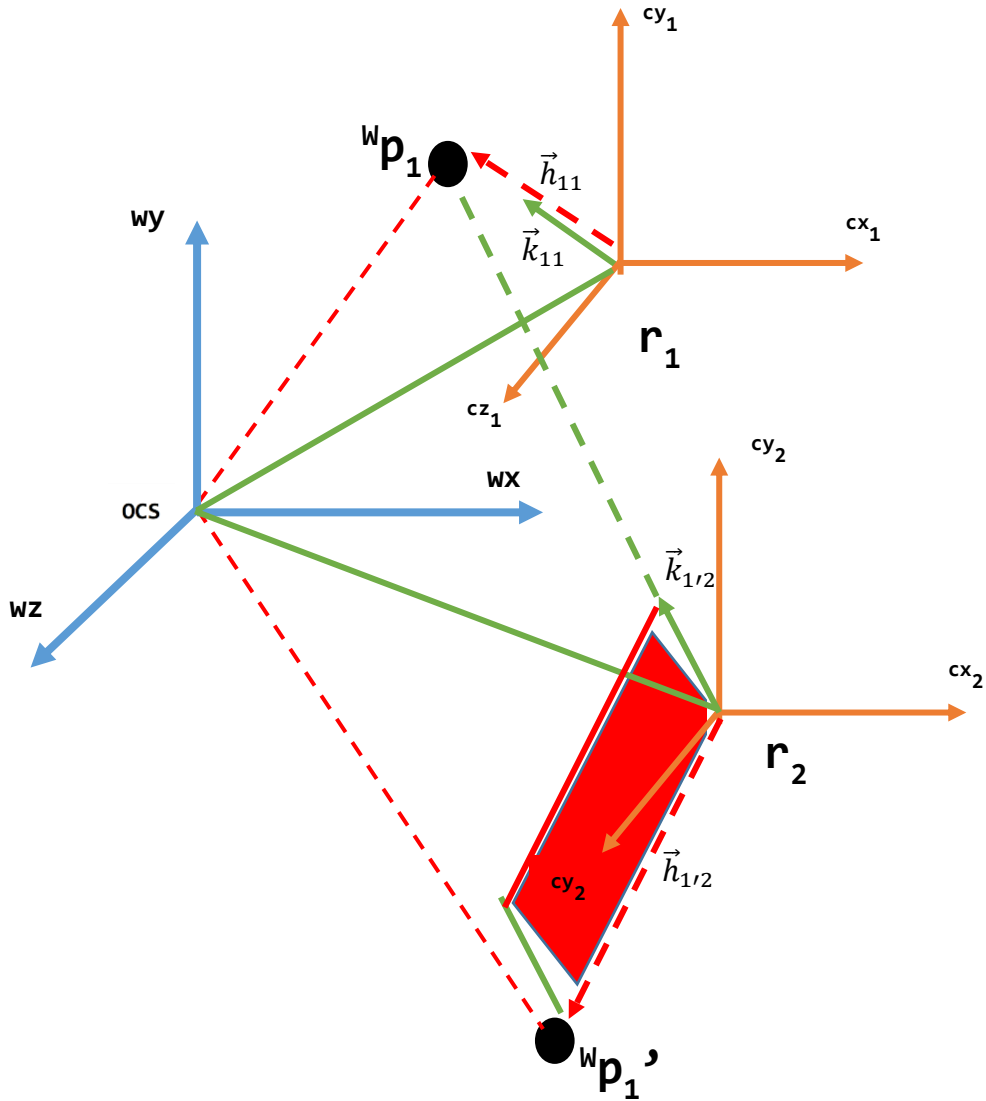
Với mỗi thời điểm có hai ảnh từ hai camera chính và phụ, hệ thống camera đồng bộ sẽ hoạt động, tìm và xác định vị trí 3D của các đặc trưng vừa tìm được bằng giải thuật Linear Triangulation Methods. Sau khi đã có vị trí 3D của các đặc trưng từ hệ thống stereo, cập nhật bản đồ lân cận với các điểm trong bản đồ là vị trí 3D của các đặc trưng tìm được sau khi đã biến đổi sang hệ tọa độ thế giới bởi vị trí robot thời điểm hiện tại vừa tính được. Các đặc trưng này được lưu với quan sát của nó nhằm tính toán cho các khâu tiếp theo.

+ Với hệ thống monocular:

Khi một đặc trưng có quan sát tương ứng tại cả hai camera (hệ thống camera đồng bộ) thì dễ dàng tính được vị trí và thêm vào bản đồ lân cận. Nhưng nhiều đặc trưng chỉ có quan sát tại camera chính (hệ thống camera đơn), việc cập nhật bản đồ lân cận cho hệ thống camera đơn chính là đi tìm vị trí 3D cho các đặc trưng như vậy dựa vào vị trí robot và quan sát của các đặc trưng các thời điểm tương ứng.

$${}^w p_i = \underset{{}^w p_i}{\operatorname{argmin}} \sum_{t \in \tau} \|({}^w p_i - r_t) \times \vec{k}_{it}\|^2 \quad (3.9)$$



- Với  $\tau$  là tập hợp các thời điểm  $t$  mà đặc trưng thứ  $i$  quan sát được trên camera chính.
- ${}^w p_i - r_t$  ký hiệu là  $\vec{h}_{it}$  là vector tính được khi có vị trí 3D của đặc trưng. Có hướng đi từ tâm camera thời điểm  $t$  đến vị trí 3D của đặc trưng  $i$ .





Hình 3-12 Minh họa các vị trí đặc trưng  $i$  thời điểm 1, 2 và sai số so với vector quan sát  $\vec{k}_{i1}, \vec{k}_{i2}$ . Vị trí có sai số nhỏ nhất (tổng diện tích hình bình hành đỏ nhỏ nhất) sẽ là vị trí đặc trưng ở tọa độ thể giới. Ở đây là vị trí  ${}^w p_1$  có sai số nhỏ nhất so với vector quan sát được.

Minh họa:

Đặc trưng thứ 1 có quan sát ở camera chính tại 2 thời điểm  $r_1$  và  $r_2$ :

+ Nếu chọn vị trí đặc trưng 1 là  ${}^Wp_1$  (đúng) thì vector quan sát được ( $\vec{k}_{11}$  và  $\vec{k}_{12}$  (màu xanh )) sẽ cùng phương với vector thực tế ( ${}^Wp_1 - r_1 = \vec{h}_{11}$  và  ${}^Wp_1 - r_2 = \vec{h}_{12}$  (màu đỏ ))  $\rightarrow$  tích vô hướng gần bằng 0 ( $S_{\text{hìnhBinhHanh}} \approx 0$ ).

+ Nếu chọn đặc trưng 1 vị trí là  ${}^Wp_1'$  (sai) thì vector quan sát ( $\vec{k}_{1/2}$  ) và dự đoán ( ${}^Wp_1' - r_2 = \vec{h}_{1/2}$  ) không cùng hướng, sẽ tạo ra tích vô hướng lớn (thể hiện qua diện tích hình bình hành đỏ).

$\rightarrow$  Chọn vị trí  ${}^Wp_1$  vì tổng tích vô hướng các quan sát được nhỏ nhất (tổng diện tích độ lỗi ( $S_{\text{hìnhBinhHanh}}$ ) bé nhất).

Chuyển về dạng tuyến tính tính  ${}^Wp_i$ :

$$\left( \sum_{t \in \tau} \mathbb{I}_3 - \vec{k}_{it} \vec{k}_{it}^T \right) {}^Wp_i = \sum_{t \in \tau} (\mathbb{I}_3 - \vec{k}_{it} \vec{k}_{it}^T) r_t \quad (3.10)$$

Với:

- $\tau$ : tập hợp các thời điểm quan sát được đặc trưng thứ  $i$  trên mặt phẳng ảnh camera chính.
- ${}^Wp_i$ : vị trí 3D đặc trưng  $i$  theo hệ tọa độ thế giới ( $W$ ).
- $r_t$ : vị trí camera thời điểm  $t$  theo ( $W$ ).

Việc cập nhật như vậy khiến hệ thống SLAM ít phụ thuộc vào đặc trưng từ hệ thống Stereo (1Hz), giúp hệ thống Stereo có thể chạy với tần số thấp hơn mà vẫn đảm bảo tính liên tục của hệ thống đồng thời khắc phục khuyết điểm của Stereo là có tầm nhìn hạn chế (chỉ tính được vị trí 3D của các điểm trong khoảng cách ngắn). Đồng thời vì kết hợp hệ thống Stereo, hệ thống SLAM còn giúp khắc phục khuyết điểm của Monocular (chỉ biết tỷ lệ khoảng cách, không biết khoảng cách chính xác).

Do các đặc trưng theo vết chưa hẳn sẽ có điểm đối sánh tại ảnh camera phụ, nên các đặc trưng có thể sẽ không có vị trí 3D tính toán được từ hệ thống camera đồng bộ. Việc cập nhật sẽ phân thành:

- Với các đặc trưng đã có vị trí 3D từ hệ thống camera đồng bộ: thêm vào vị trí 3D từ hệ thống monocular.
- Với các đặc trưng chưa có vị trí 3D từ camera đồng bộ: chỉ thêm các đặc trưng vào bản đồ nếu số lượng các đặc trưng trong bản đồ xuống dưới một ngưỡng cho phép.

### 3.4.2 Xóa đặc trưng:

Nếu hệ thống theo vết trên chuỗi ảnh từ camera chính bị mất dấu đặc trưng (lost tracking), những đặc trưng như thế sẽ không còn có quan sát (từ ảnh) vì vậy sẽ được xóa khỏi bản đồ lân cận và thêm vào những đặc trưng mới được phát hiện nhằm bảo đảm tính liên tục cho hệ thống.

### 3.5 Phát hiện lỗi trong bản đồ lân cận:

Vì vị trí 3D đặc trưng được tính toán từ hai cách, một là từ theo vết một camera đơn (Monocular) và từ đặc trưng so khớp hai camera đồng bộ (Stereo) nên khi hệ thống tích lũy lỗi trong quá trình ước lượng có thể được phát hiện thông qua vị trí 3D tương quan được ước lượng từ hai hệ thống. Sai số ước lượng tại thời điểm  $t$  được tính thông qua:

$$\gamma = \frac{1}{|\mathcal{K}|} \sum_{k \in \mathcal{K}} \frac{\|w p_k^m - r_j\|}{\|w p_k^s - r_j\|} \quad (3.11)$$

Với:

- $w p_k^s$ : vị trí 3D được tính từ so khớp hai camera đồng bộ.
- $w p_k^m$ : vị trí 3D được tính từ theo vết camera đơn.

Nếu hai hệ thống đồng bộ và hoạt động tốt,  $\gamma \approx 1$  và ngược lại sẽ không đồng bộ và có lỗi trong bản đồ lân cận.

Sau khi phát hiện lỗi, hệ thống sẽ loại bỏ mọi đặc trưng nhận được từ hệ thống một camera đơn và khởi động lại bản đồ lân cận dựa vào những vị trí 3D từ camera đồng bộ  $w p_k^s$ .

### 3.6 Kết hợp với IMU qua lọc nhiễu Kalman UKF (Unscented Kalman filter):

Hệ thống chỉ dựa vào đặc trưng thị giác sẽ cho ra đầu ra vị trí robot với tần số 25Hz bằng với tần số ảnh đến từ camera chính (25 frame/s). Tần số đầu ra này là chưa đủ để điều khiển robot một cách hiệu quả, chính vì thế hệ thống kết hợp thêm bộ lọc Kalman UKF để bổ sung ước lượng vị trí và vận tốc robot khi đặc trưng thị giác vẫn còn đang trì hoãn. Đầu ra cuối cùng sau khi qua bộ lọc Kalman UKF là 100 output/s.

Vector trạng thái của hệ thống được định nghĩa:

$$X = [s, \dot{s}, \Phi, a_b, \Psi_b]^T \quad (3.12)$$

Trong đó:

- $s = [wx, wy, wz]^T$ : vị trí camera trong hệ trục tọa độ thế giới WCS.
- $\Phi = [\phi, \theta, \psi]^T$ : góc xoay roll, pitch, yaw thể hiện hướng xoay của robot.
- $a_b = [a_{bx}, a_{by}, a_{bz}]^T$ : giá trị thêm vào (bias) của gia tốc robot trong không gian 3D.
- $\Psi_b = [\phi_b, \theta_b]^T$ : giá trị thêm vào (bias) của góc xoay roll, pitch từ IMU.

Mô hình tính toán: Mô hình tính toán dựa trên IMU:

$$\begin{aligned} u &= [\omega, a]^T = [\omega_x, \omega_y, \omega_z, a_x, a_y, a_z]^T \\ v &= [v_\omega, v_a, v_{ab}]^T \\ X_{t+1} &= f(X_t, u_t, v_t) \end{aligned} \quad (3.13)$$

Trong đó:

- $u$ : vận tốc góc (rad/s) và gia tốc theo các chiều x, y, z từ IMU.
- $v$ : biểu diễn nhiễu Gaussian liên kết với vận tốc góc, gia tốc và giá trị thêm vào của gia tốc.

Mô hình đo lường:

Vị trí robot tính toán được từ hệ thống thị giác sẽ được chuyển sang vị trí IMU và kết hợp với góc xoay từ IMU để tạo thành đo lường vị trí 6-bậc-tự-do (6-DOF):

$$Z = \begin{bmatrix} r + R_I^W R_C^I d_C^I \\ \Phi \end{bmatrix} \quad (3.14)$$

Trong đó:

- $d_C^I$  : là phép dịch chuyển từ camera chính sang IMU. (vị trí IMU và camera chính là đã biết)

Mô hình đo lường là mô hình tuyến tính và được viết dưới dạng:

$$Z = HX + n \quad (3.15)$$

Trong đó:

- $H$  : ma trận trích xuất vị trí 6-bậc-tự-do (6-DOF) trong trạng thái và  $n$  là nhiễu Gaussian thêm vào.
- Vì mô hình đo lường là tuyến tính nên có thể cập nhật đo lường thông qua bước cập nhật của bộ lọc Kalman tuyến tính.

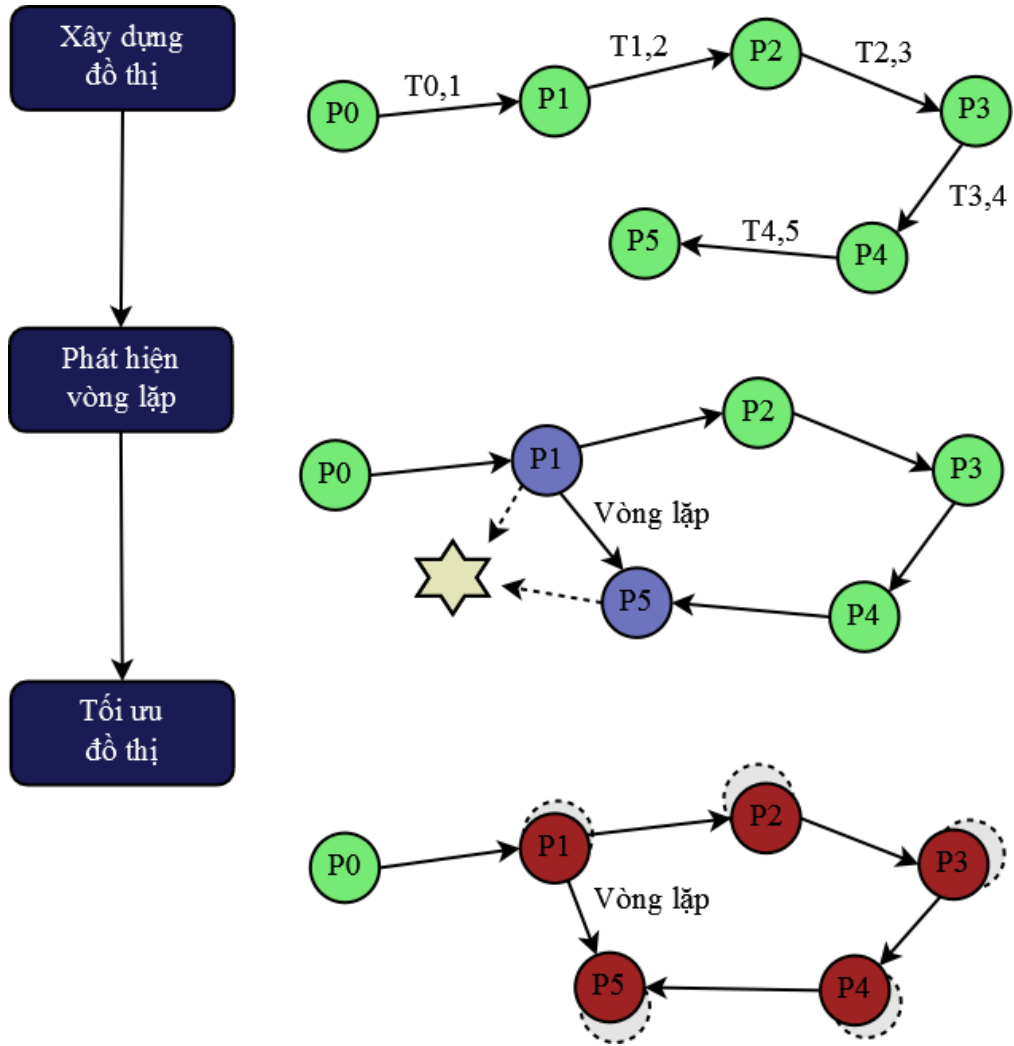
## CHƯƠNG 4: TỐI ƯU ĐỒ THỊ VÀ XÂY DỰNG BẢN ĐỒ

*Trong chương trước, ta đã biết được làm cách nào để ước lượng vị trí robot và hướng xoay dựa vào thông tin từ stereo camera và IMU. Ở chương này, ta sẽ bàn đến việc làm cách nào để từ những phép biến đổi đã ước lượng giữa hai khung hình mà xây dựng được đồ thị biểu diễn cho bài toán SLAM. Khi robot quay lại các nơi đã từng đi qua, trong đồ thị sẽ xuất hiện các vòng lặp và tối ưu đồ thị trên những vòng lặp này sẽ cho vị trí tối ưu của robot. Và khi biết vị trí tối ưu, một bản đồ ba chiều cuối cùng sẽ được xây dựng.*

### 4.1. Mô hình hóa bài toán SLAM bằng đồ thị:

Sau khi biết được chuyển động ước lượng từ hệ tọa độ thế giới đến thời điểm hiện tại, ta có thể biết được chuyển động giữa hai thời điểm liên tiếp nhau. Dựa vào chuyển động giữa hai thời điểm, ta có thể mô hình hóa lại bài toán SLAM bằng một đồ thị. Trong đồ thị này, mỗi đỉnh sẽ tương ứng với một vị trí của robot tại một thời điểm và mỗi cạnh sẽ là phép biến đổi ước lượng được giữa hai vị trí này. Khi robot chưa quay lại vị trí trước, đồ thị sẽ chỉ tuyến tính, với mỗi đỉnh chỉ có một cạnh nối đến đỉnh tiếp theo. Khi robot quay trở lại một vị trí trước đó, ta có thể tìm được mối quan hệ giữa đỉnh hiện tại và một đỉnh trong quá khứ mà khi thêm cạnh giữa hai đỉnh này sẽ tạo thành một vòng lặp. Do trong quá trình tính toán, sai số tích lũy theo thời gian, nên khi có vòng lặp, đồ thị sẽ không còn thống nhất với nhau. Vì thế cần có một quá trình để tối ưu đồ thị, tìm quan hệ vị trí mới và cũ để chỉnh sửa lỗi trong quá trình ước lượng, sử dụng đồ thị để tối ưu các vị trí trong quá trình di chuyển. Quá trình này gồm các bước: xây dựng đồ thị, phát hiện vòng lặp và tối ưu đồ thị.





Hình 4-1 Quá trình xây dựng và tối ưu đồ thị

#### 4.1.1 Tìm phép chuyển động giữa hai thời điểm:

Với việc chuyển động giữa hệ tọa độ thế giới và thời điểm hiện tại đã biết (gồm phép xoay và phép tịnh tiến). Gọi  $F_{t-1,t}$  là phép biến đổi đã ước lượng giữa hai thời điểm liên tiếp.  $F_{t-1,t}$  sẽ bao gồm:

$$F_{t-1,t} = [R_{t-1,t} \quad | \quad T_{t-1,t}] \quad (4.1)$$

Với  $R_{t-1,t}$  là phép xoay từ thời điểm  $t-1$  đến thời điểm  $t$ .  $T_{t-1,t}$  là phép dịch chuyển từ  $t-1$  đến  $t$ .  $R_{t-1,t}$  có thể được tính nhờ phép xoay  $R_{t_t}^W$  và  $R_{t_{t-1}}^W$  là hai ma trận

phép xoay từ thời điểm  $t$  và  $t-1$  đến hệ tọa độ thế giới (có được từ IMU tại từng thời điểm). Khi đó, phép xoay từ thời điểm  $t-1$  đến  $t$  sẽ được tính:

$$\text{Vì: } R_{I_{t-1}}^W = R_{I_t}^W R_{I_{t-1}}^{I_t}$$

$$\text{Nên: } R_{I_{t-1}}^{I_t} = (R_{I_t}^W)^{-1} R_{I_{t-1}}^W = R_W^{I_t} R_{I_{t-1}}^W \quad (4.2)$$

$$\text{Và phép dịch chuyển từ } T_{t-1, t} \text{ sẽ được tính: } T_{t-1, t} = r_t - r_{t-1} \quad (4.3)$$

#### 4.1.2 Xây dựng đồ thị:

Một cách tiếp cận là mỗi thời điểm đều thêm vào đồ thị thể hiện cho bài toán SLAM tương ứng đỉnh là vị trí robot mỗi thời điểm và cạnh nối giữa hai đỉnh là phép biến đổi  $F$  giữa hai thời điểm. Tuy nhiên, một số trường hợp, robot có thể di chuyển chậm hoặc ngừng di chuyển, khiến cho vị trí có ít sự thay đổi. Vì thế việc đưa tất cả những thời điểm này khiến đồ thị dư thừa và làm chậm quá trình tối ưu đồ thị. Ngoài ra, như đã đề cập ở mục 3.3.1, vị trí ước lượng được từ thuật toán RANSAC không phải là tối ưu. Khi ước lượng vị trí có sai số sẽ dẫn đến việc xác định vị trí 3D đặc trưng sai số và cộng dồn lại cho ra vị trí hiện tại tích tụ sai số gây ra hiện tượng “trôi” (drift). Chính vì vậy, chỉ đưa vào các vị trí quan trọng, tức là những vị trí tại đó có sự thay đổi trong không gian lớn trong quá trình di chuyển của robot vào đồ thị như là các đỉnh. Vị trí của robot sẽ được tối ưu dựa trên tìm mối quan hệ giữa đỉnh mới nhất với những đỉnh quan trọng như vậy và tối ưu toàn cục đồ thị. Việc này góp phần làm giải quyết hiện tượng sai số tích lũy. Vì thế cần có một cách để xác định những vị trí quan trọng có thể giúp cho quá trình tối ưu này:

Nếu đây là vị trí quan trọng thì phải là một thời điểm có ảnh từ cả hai camera (Stereo). Ta chỉ sử dụng những điểm 3D từ mô hình hai camera để chống nhiễu trong quá trình ước lượng vị trí 3D của những điểm môi trường vốn dựa trên các vị trí ước lượng đã có sai số bên trong. Và vị trí đó có một trong hai điều kiện sau xảy ra:

- Chuyển động xoay hoặc tịnh tiến trong không gian lớn. Đỉnh mới sẽ được thêm vào nếu khoảng cách tịnh tiến hoặc góc quay trong phép biến đổi giữa hai thời điểm liên tiếp vượt quá một ngưỡng cho trước. Việc đặt ra một ngưỡng

như vậy giúp cho các đỉnh có sự khác nhau về vị trí, tránh tình trạng mọi đỉnh đều gần như nhau làm dư thừa và chậm cho việc tối ưu đồ thị.

- Số lượng điểm tương ứng thu được sau quá trình theo vết giữa hai thời điểm liên tiếp trên ảnh camera chính nhỏ hơn một ngưỡng. Ý nghĩa của điều kiện này là có những trường hợp ta theo vết được ít đặc trưng từ ảnh dẫn đến tìm được ít các điểm tương ứng. Lúc này phải có một sự biến động vị trí trong không gian robot nên việc đưa đỉnh như vậy vào đồ thị là hợp lý.

Khi thỏa mãn một trong hai điều kiện trên, một đỉnh và một cạnh nối từ đỉnh đến đỉnh trước đó sẽ được thêm vào.

### 4.1.3 Phát hiện vòng lặp:

Khi quay lại vị trí cũ, sẽ xuất hiện mối liên hệ giữa đỉnh hiện tại và các đỉnh trước đó trong đồ thị, ta cần phải tìm ra các mối quan hệ này và thêm vào đồ thị như là vòng lặp. Vì thế, việc phát hiện vòng lặp được thực hiện bằng cách ước lượng chuyển động giữa đỉnh mới thêm vào và các đỉnh trong đồ thị, nếu thuật toán ước lượng chuyển động hai thời điểm cho kết quả là một chuyển động tốt, một cạnh mới sẽ được thêm vào giữa hai đỉnh đang xét. Vòng lặp được phát hiện nếu có cạnh mới được thêm vào.

Vì có nhiều đỉnh trong đồ thị mà ta không thể nào tìm phép biến đổi cho đỉnh mới nhất với tất cả đỉnh như vậy, dễ dàng thấy là phương pháp này sẽ rất tốn thời gian. Với  $N$  đỉnh, cần kiểm tra tổng cộng là  $N(N - 1)/2$  lần, việc này sẽ khiến hệ thống quá tải dẫn đến không đáp ứng được thời gian phản ứng thực tế khi số lượng đỉnh ngày càng tăng.

#### Ước lượng chuyển động giữa hai khung hình:

Sau bước đối sánh đặc trưng, ta sẽ được một tập điểm tương ứng giữa hai khung hình. Tuy nhiên, ta chỉ biết được vị trí hai chiều trên ảnh của các điểm tương ứng. Với sự xuất hiện của dữ liệu độ sâu, ta có thể biết được vị trí ba chiều dựa vào việc xác định vị trí 3D cho đặc trưng thực hiện từ mô hình camera đồng bộ. Lúc này ta sẽ có tập điểm  $S = \{s = (X, Y, Z, 1)^T\}$  và tập điểm  $S' = \{s' = (X', Y', Z', 1)^T\}$ ,

trong đó  $s$  và  $s'$  biểu diễn cho cùng một điểm ba chiều trên thực tế nhưng được tính theo hai trục tọa độ thuần nhất khác nhau.  $s$  được tính theo trục tọa độ của camera chính tại thời điểm  $t-1$ :  $r_{t-1}$  còn  $s'$  được tính theo tại thời điểm  $t$ :  $r_t$ .

Từ những cặp điểm tương ứng đã tìm được ở bước trước, ta có thể tìm được chuyển động từ vị trí  $r_{t-1}$  đến vị trí  $r_t$ . Phép biến đổi này có thể được biểu diễn bằng một ma trận  $F_{t-1}^t$  có kích thước  $4 \times 4$  gồm phép xoay  $R$  và tịnh tiến  $T$  giữa hai vị trí camera chính. Ta có thể biến đổi một điểm  $s = (X, Y, Z, 1)^T$  thuộc hệ trục tọa độ tại vị trí  $r_{t-1}$  sang điểm  $s' = (X', Y', Z', 1)^T$  thuộc hệ trục tọa độ tại vị trí  $r_t$  dựa vào công thức sau:

$$s' = Fs \quad (4.4)$$

Phép biến đổi cần tìm là phép biến đổi thỏa mãn phép biến đổi  $F$  này cho nhiều cặp điểm nhất. Bài toán này có thể giải bằng phương pháp bình phương tối thiểu với tất cả các cặp điểm, với mỗi cặp điểm tương ứng với một phương trình. Tuy nhiên, chỉ cần một điểm tương ứng tìm được là sai hoặc nhiều đến từ Stereo camera là có thể dẫn đến chuyển động ước lượng được là không chính xác.

Để xử lý nhiễu trong tập điểm tương ứng, khóa luận sử dụng phương pháp RANSAC để ước lượng chuyển động. Ở mỗi lần lặp, một tập mẫu nhỏ vừa đủ ước lượng sẽ được chọn ngẫu nhiên từ các cặp điểm. Sau khi tìm phép biến đổi ước lượng được từ tập mẫu và đánh giá kết quả ước lượng dựa trên một hàm độ lỗi. Các bước trên được lặp lại nhiều lần với mỗi lần ta chọn ra một tập mẫu mới và giữ lại phép biến đổi tốt nhất. Việc sử dụng RANSAC sẽ giúp loại bỏ nhiễu trong quá trình lấy ngẫu nhiên cặp điểm và tìm sai số bé nhất như vậy. Những điểm thỏa mãn phép biến đổi RANSAC tốt nhất đó sẽ dùng để tính toán phép biến đổi cuối cùng.

## **Giải thuật 2** Tìm phép biến đổi ba chiều với RANSAC

**Đầu vào:** Tập điểm ba chiều  $S = \{s = (X, Y, Z, 1)^T\}$  và  $S' = \{s' = (X', Y', Z', 1)^T\}$

**Đầu ra:** Phép biến đổi để chuyển từ  $S$  sang  $S'$  được biểu diễn bởi ma trận  $F_{(3 \times 4)}$

**Mã giả:**

$bestTransform, bestInliers \leftarrow \emptyset$

**for**  $iteration = 1$  to  $N$  **do**

$samples \leftarrow$  chọn ngẫu nhiên  $k$  cặp điểm từ  $S$  và  $S'$  theo xác suất

Tính phép biến đổi  $currentTransform$  từ tập  $samples$

$inliers \leftarrow \emptyset, rmse \leftarrow 0$

**for each**  $P_i \in S$  **do**

$error \leftarrow \|currentTransform \cdot s_i - s'_i\|^2$

**if**  $error < threshold$  **then**

$inliers \leftarrow inliers + (s_i, s'_i)$

$rmse \leftarrow rmse + error$

**end if**

**end for**

$rmse \leftarrow \sqrt{rmse/|inliers|}$

**if**  $|inliers| > requiredInliers$  **then**

Tính lại phép biến đổi  $currentTransform$  từ tập  $inliers$

Tính lại tập  $inliers$  từ phép biến đổi  $currentTransform$

**if**  $currentTransform$  tốt hơn  $bestTransform$  **then**

$bestTransform \leftarrow currentTransform$

$bestInliers \leftarrow inliers$

**end if**

**end if**

**end for**

**return**  $bestTransform, bestInliers$

---

Để tính phép biến đổi giữa hai thời điểm, ta sử dụng thuật toán của Huang [1] đề xuất. Về cơ bản thuật toán sẽ tìm phép biến đổi  $F$  sao cho tối thiểu hàm sai số biến đổi giữa hai cặp điểm:

$$\arg \min_F \sum_{i=1}^k \|Fs_k - s'_k\|^2 \quad (4.5)$$

Với  $k$  là số điểm tương ứng và tối thiểu là 3. Các bước tính toán phép biến đổi ba chiều:

- Tính điểm trung bình  $\bar{s}$  của tập  $S$  và  $\bar{s}'$  của tập  $S'$ .
- Tính ma trận hiệp phương sai  $H = \sum_{i=1}^k (s_i - \bar{s})(s'_i - \bar{s}')^T$ .
- Sử dụng phân tích giá trị đơn (Singular Value Decomposition – SVD) cho ma trận  $H = USV^T$ .
- Ma trận phép xoay sẽ  $R = VU^T$ .
- Phép tịnh tiến được tính bởi  $T = \bar{s}' - R\bar{s}$ .

Trong quá trình lấy ngẫu nhiên RANSAC, số điểm tương ứng tối thiểu cho số lượng mẫu lấy ở mỗi vòng lặp (3 mẫu) sẽ được sử dụng.

Để đánh giá độ lỗi của phép biến đổi vừa ước lượng được, sẽ dựa vào:

- Số lượng cặp điểm thỏa phép biến đổi. (inliers)
- Căn bậc hai trung bình bình phương sai số (*root mean squared error*) khi biến đổi vị trí các điểm từ tập  $S$  sang tập  $S'$  sử dụng F. Sai số càng lớn, độ lỗi phép biến đổi cũng càng lớn.

Vì quá trình tính sai số có thể chỉ tính những điểm thuộc phép biến đổi mà những điểm đó có thể là nhiễu, dù sai số ít nhưng phép biến đổi đó vẫn không tốt. Việc so sánh chất lượng giữa hai phép biến đổi dựa vào hai tiêu chí này. Phép biến đổi nào hơn ở cả hai tiêu chí thì mới được xem là phép biến đổi tốt hơn.

Vì vấn đề hiệu suất, phải giảm các đỉnh cần kiểm tra. Khóa luận sẽ chọn ngẫu nhiên ra một tập đỉnh (ví dụ 20 đỉnh) trong tất cả các đỉnh của đồ thị (trừ các đỉnh gần nhất) để làm ứng cử viên phát hiện vòng lặp. Để giúp quá trình phát hiện vòng lặp, một cạnh được nối tới một đỉnh, quá trình tiếp theo sẽ tìm kiếm ở những đỉnh lân cận đỉnh vừa tìm được, nếu cho ra cạnh tốt hơn thì cạnh tốt hơn ấy sẽ được thêm vào thay cho cạnh vừa thêm trước đó.

#### 4.1.4 Tối ưu đồ thị:

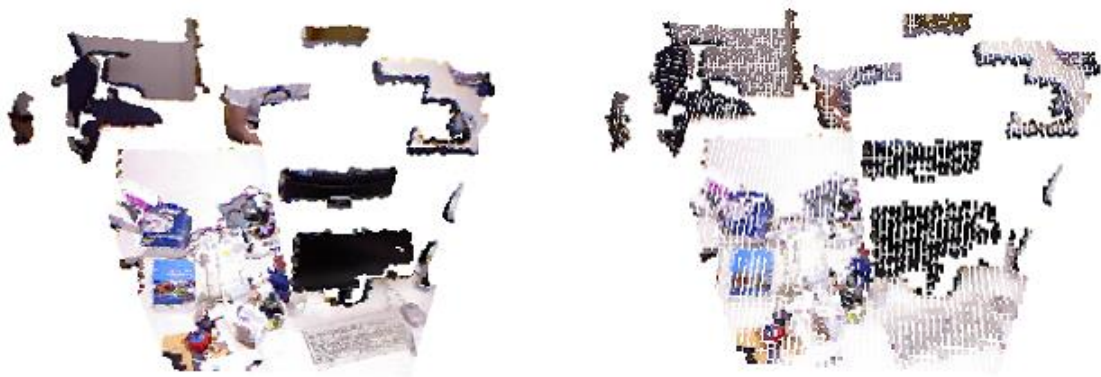
Mỗi khi thêm vào vòng lặp, đồ thị lúc này sẽ xuất hiện sự không thống nhất về vị trí của các đỉnh và các cạnh giữa đỉnh tức là phép biến đổi và vị trí robot các thời điểm. Quá trình tối ưu sẽ được khởi động, việc này là việc giải một bài toán bình phương tối thiểu phi tuyến. Gọi  $E = \{F_{t-1}^t\}$  là tập tất cả các cạnh trong đồ thị và  $P = \{r_t\}$  là tập tất cả các đỉnh trong đồ thị, ta có hàm chi phí sau cần phải tối ưu:

$$A(P) = \sum_{T_{ij} \in E} \|F_{t-1}^t r_{t-1} - r_t\|^2 \quad (4.6)$$

Vì phép biến đổi  $F$  có bao gồm phép xoay (phi tuyến) nên cần phải sử dụng các thuật toán tối ưu hàm phi tuyến. Thuật toán Levenberg-Marquardt sẽ được sử dụng để tối ưu tham số trên dựa trên hàm chi phí. Sau khi đồ thị đã được tối ưu, vị trí hiện tại của robot cũng sẽ được cập nhật dựa trên vị trí của đỉnh mới nhất sau khi đã được tối ưu.

#### 4.2 Xây dựng bản đồ

Khi toàn bộ vị trí của robot đã được tối ưu, ta có thể tái tạo lại bản đồ ba chiều. Với mỗi khung hình, ta đã có thể có được một đám mây điểm ba chiều được tái tạo từ hai camera bằng cách tái tạo độ sâu từng cặp điểm tương ứng trong hai camera. Tuy nhiên, để tái tạo lại toàn bộ bản đồ ba chiều tất cả thời điểm, các đám mây điểm này vẫn đang được tính trên tọa độ camera tương ứng của chúng. Vì vậy ta cần phải đưa các đám mây điểm này về một hệ trục tọa độ chung. Do ta đã biết được vị trí của camera, mỗi đám mây điểm sẽ được biến đổi dựa theo vị trí camera để cho ra đám



*Hình 4-2 Đám mây điểm ban đầu và đám mây điểm sau khi đã sử dụng bộ lọc lưới voxel (phải). Tuy kích thước của đám mây điểm đã giảm nhưng vẫn thể hiện rõ môi trường.*

mây điểm theo hệ trục thế giới. Các đám mây điểm đã được biến đổi sau đó sẽ ghép với nhau để tạo thành bản đồ ba chiều cuối cùng.

Tùy từng loại camera mà cho ra ảnh với kích thước khác nhau, nhưng một chuẩn thường sử dụng là 640x480. Vì thế số lượng điểm trong đám mây điểm cũng tương đương con số này. Sử dụng toàn bộ điểm trong đám mây điểm này để biểu diễn bản đồ là không cần thiết. Vì vậy, ở đây khóa luận sử dụng bộ lọc lưới voxel để giảm kích thước đám mây điểm. Phương pháp này chia đám mây điểm thành những khối vuông bằng nhau, tất cả các điểm trong một phân ô vuông sẽ được thay thế bằng điểm trọng tâm của các điểm trong phần đó. Đám mây điểm giảm kích thước và sẽ được biến đổi sang tọa độ thế giới và ghép với nhau để cho ra bản đồ chung.

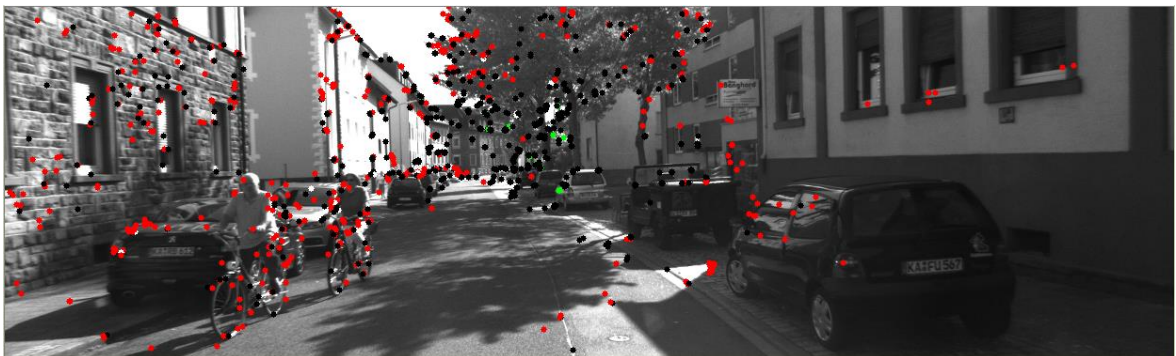


# CHƯƠNG 5: PHÁT HIỆN VÀ NÉ TRÁNH VẬT CẢN

*Trong chương này sẽ bàn đến vấn đề sau khi đã có vị trí và bản đồ 3D xung quanh robot, để robot có thể tự hành thì robot phải có khả năng phát hiện và né tránh các vật cản đến từ môi trường. Vì robot có thể di chuyển nhanh trong môi trường nên cần khả năng phát hiện vật cản nhanh tức là tìm được khoảng cách từng thời điểm đến vật và kích thước 3D của vật cản.*

## 5.1 Phát hiện vật cản:

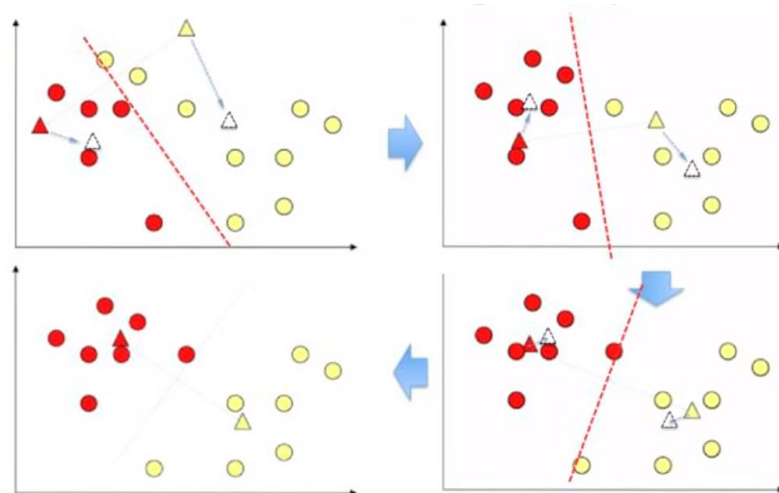
Vật cản được nói ở đây là vật cản tĩnh trong môi trường. Tại mục 3.4.1, tại mỗi thời điểm, sẽ cập nhật bản đồ lân cận vị trí 3D của các đặc trưng. Vì quá trình cập nhật là liên tục tại mỗi thời điểm và các đặc trưng chủ yếu đến từ các vật thể trong môi trường (xe cộ, nhà cửa, ...). Nên khi một số lượng đặc trưng lớn tập trung, tức là có vị trí 3D gần với nhau trong hệ tọa độ thế giới, ta có thể xem xét nó là vật cản và điều hướng robot tránh khỏi vật cản.



*Hình 5-1 Các điểm màu đỏ là các đặc trưng đã có vị trí 3D xác định bởi hệ thống một camera, ta có thể thấy các đặc trưng tập trung chủ yếu tại các vật thể (xe cộ, cây cối trên đường và ít tập trung tại mặt đường và hoàn toàn không có trong không khí)*

Vì hệ trục thế giới cũng là hệ trục IMU dựa vào để cho các giá trị roll, pitch, yaw (góc xoay so với hệ trục đó) có trục z vuông góc với mặt phẳng đường (do IMU dựa vào trọng lực trái đất để xác định góc xoay)- Vì khoảng cách vật thể ở quá xa thì robot tạm thời chưa cần quan tâm, nên ở đây khóa luận cũng sẽ không xét các đặc trưng có vị trí 3D ở quá xa lớn hơn một ngưỡng (ở đây là 10m) so với vị trí robot thời điểm hiện tại trong hệ trục thế giới.

Sau khi loại đi những điểm không cần quan tâm, khóa luận sẽ gom nhóm những điểm 3D còn lại trong bản đồ lân cận theo vị trí 3D của chúng, thuật toán gom nhóm là “k-điểm trọng tâm” (k-means), ý tưởng chính thuật toán là các điểm trọng tâm của từng nhóm phải bền vững qua các vòng lặp, mỗi vòng lặp sẽ tính lại các điểm trọng tâm bằng các điểm có khoảng cách gần trọng tâm lần lặp trước nhất, vì vậy nếu các điểm trọng tâm thực sự sẽ bền vững qua các vòng lặp.



*Hình 5-2 Minh họa thuật toán k-means, với điểm trọng tâm là tam giác. Lần lặp đầu tiên sẽ random ngẫu nhiên k điểm trọng tâm, các điểm còn lại sẽ xếp vào nhóm của điểm trọng tâm có khoảng cách gần nhất, sau đó thuật toán sẽ tính lại các điểm trọng tâm. Vòng lặp tiếp theo cũng sẽ thực hiện các bước tương tự cho đến khi các điểm trọng tâm không còn thay đổi.*

Nếu một nhóm có trên số lượng điểm nhất định thì sẽ được xem xét là vật cản. Vị trí vật cản sẽ là trọng tâm của các điểm đặc trưng thuộc vật cản và kích thước là

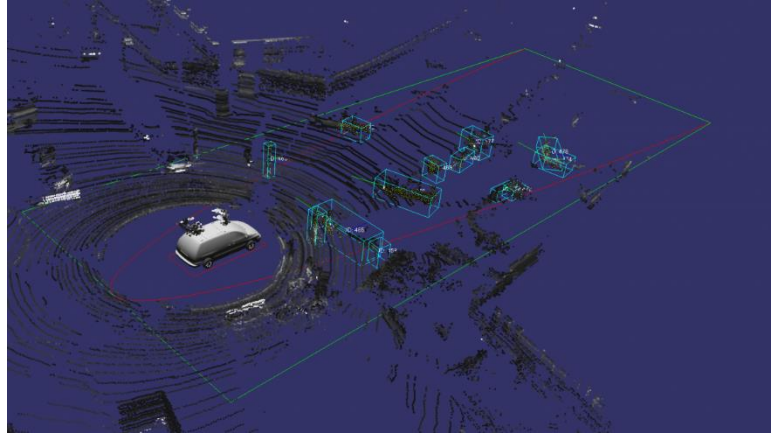
khung bao của các điểm 3D đó, tức là khung 3D nhỏ nhất có thể chứa tất cả tập điểm trong nhóm (tính thông qua các giá trị  $w_x$ ,  $w_y$ ,  $w_z$  lớn nhất của các điểm trong nhóm).

## 5.2 Né tránh vật cản:

Ở pha né tránh vật cản, khóa luận chỉ đề xuất một mô hình có thể tránh né các vật thể tĩnh và với đường đi robot trong không gian 2D (tức là robot di chuyển trên mặt phẳng không thể bay lên cao). Khóa luận dựa trên phương pháp “theo khoảng trống” (follow the gap [17]), phương pháp này là gặp vật cản thì phản ứng lại với vật cản mà không xét đến việc đến đích, còn nếu không có vật cản sẽ xét đến việc đến đích. Mục đích là đặt yếu tố an toàn của robot lên cao nhất bằng cách né tránh tốt vật cản trước mắt và đường đi của robot cũng chính là đường thẳng nối vị trí robot và đích đến.

Về robot: từ pha SLAM, ta đã có thông tin vị trí robot thời điểm hiện tại và hướng xoay của robot trong hệ trục thế giới.

Về vật cản: từ việc phát hiện vật cản trước đó, ta đã có thông tin về vật cản là khoảng cách từ vật cản đến robot và kích thước thể hiện qua khung bao của vật cản trong hệ trục tọa độ thế giới. Để dễ dàng trong việc né tránh vật cản, các vật cản sẽ được chuyển về hệ tọa độ IMU thời điểm hiện tại, bằng cách sử dụng phép xoay từ hệ tọa độ thế giới đến IMU thời điểm hiện tại  $R_W^{I_t}$  và phép tịnh tiến từ hệ tọa độ thế giới đến robot cũng là vị trí robot  $r_t$ :  $Obstacle_i^{Robot} = R_W^{I_t} Obstacle_i^W - r_t$  (5.1)



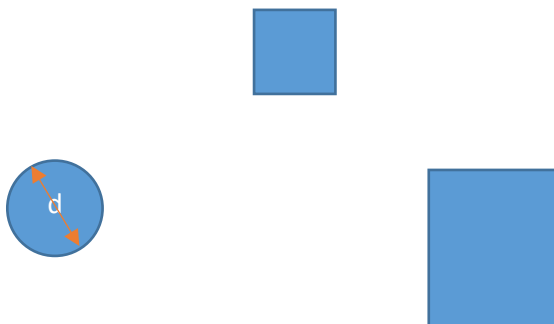
*Hình 5-3 Ví dụ về khung bao 3D bao quanh vật cản đã phát hiện. Vị trí vật cản (tâm khung bao) đã biết trong pha phát hiện vật cản và vị trí robot (ở đây là chiếc xe màu trắng) đã có trong pha SLAM.*

Mục tiêu phương pháp này ưu tiên độ an toàn nên ý tưởng của phương pháp là tìm khoảng trống lớn nhất trước mắt để đi. Vì thế mỗi vật cản cần tìm góc biên trái và phải sau đó tìm góc trống lớn nhất mà đi. Tuy nhiên, để xét thêm yếu tố đến đích nên hệ thống sẽ kết hợp thêm hướng đến đích vào hướng đến góc lớn nhất. Đồng thời xét thêm đến yếu tố khoảng cách đến vật cản, vật ở xa sẽ ưu tiên đến đích và vật ở gần sẽ ưu tiên né tránh vật cản hơn việc đến đích, điều này nhằm tránh khỏi việc né tránh quá sớm lệch xa khỏi đích ban đầu không cần thiết và việc lo né tránh không quan tâm đến đích sẽ dễ bị lạc hướng (lo đi đến khoảng trống lớn nhất không bao giờ đến đích).

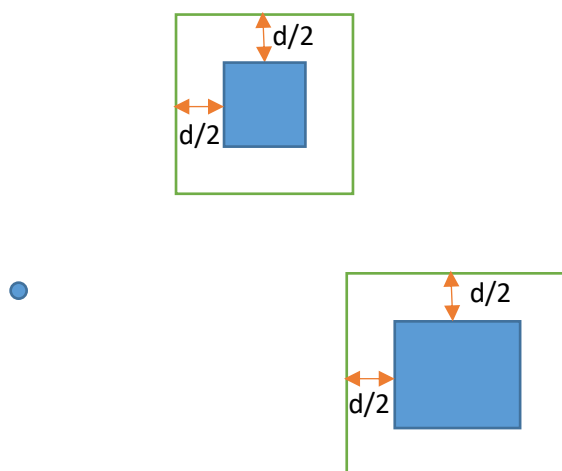
Phương pháp né tránh vật cản theo khoảng trống gồm có 5 bước: Bước 1: Mở rộng kích thước vật cản. Bước 2: Biểu diễn vật cản. Bước 3: Tạo danh sách khoảng trống và tìm khoảng trống lớn nhất. Bước 4: Tính góc trung tâm khoảng trống lớn nhất. Bước 5: Xác định hướng di chuyển cho robot có xét góc đến đích.

### **5.2.1 Mở rộng kích thước vật cản**

Để không phải xét đến kích thước robot, khóa luận sẽ tiến hành mở rộng kích thước vật cản, kích thước mỗi vật cản mở rộng sẽ tương đương với kích thước robot, tức là mỗi chiều vật cản sẽ được tăng thêm  $\frac{1}{2}$  kích thước robot.



Hình 5-4 Robot (tròn) và vật cản ( chữ nhật )

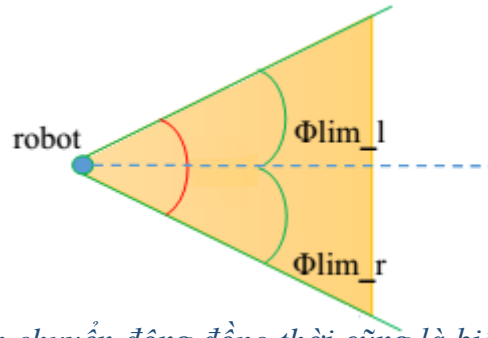


Hình 5-5 Robot dạng điểm và vật cản sau khi mở rộng kích thước

## 5.2.2 Biểu diễn vật cản

### 5.2.2.1 Biên chuyển động

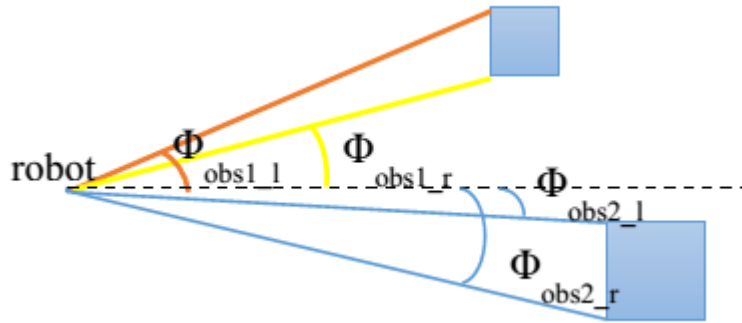
Biên chuyển động là giới hạn đặt ra cho robot để thực hiện phép xoay, nếu biên chuyển động đặt ra quá lớn thì góc lớn nhất luôn nằm ở biên trái hoặc phải và robot sẽ luôn đi ngược hướng đến đích. Vì thế, góc chuyển động được đặt là góc quan sát được của camera nhằm mục đích giới hạn góc chuyển động đồng thời dù robot phải xoay góc lớn vẫn nằm trong tầm bản đồ đã quan sát được.



Hình 5-6 Biên chuyển động đồng thời cũng là biên quan sát thiết bị

### 5.2.2.2 Tính góc vật cản

Mỗi vật cản sẽ biểu diễn bởi hai thông số đó là góc biên trái và góc biên phải của vật cản



Hình 5-7 Góc biểu diễn vật cản,  $\phi_{obs1_l}$  và  $\phi_{obs1_r}$  là góc biên trái và phải của vật cản thứ 1

Vì hệ trục IMU dựa vào để cho các giá trị roll, pitch, yaw (góc xoay so với hệ trục đó) có trục z vuông góc với mặt phẳng đường (do IMU dựa vào trọng lực trái đất để xác định góc xoay). Nên có thể xác định các góc  $\phi$  như sau:

$$\phi_{obsi_l} = \arctan \left( \frac{\phi_{obsi_l.x}}{\phi_{obsi_l.y}} \right) \quad (5.2)$$

$$\phi_{obsi_r} = \arctan \left( \frac{\phi_{obsi_r.x}}{\phi_{obsi_r.y}} \right)$$

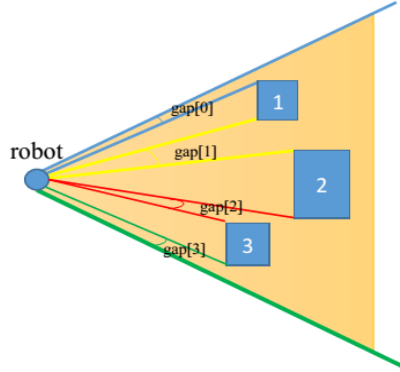
Với:  $\phi_{obsi_l}$ : góc biên trái của vật cản thứ i

$\phi_{obsi_r}$ : góc biên phải của vật cản thứ i

$\phi_{obsi_l.x}$ : là tọa độ x của điểm trái cùng của vật cản thứ i.

$\phi_{obsi\_l,y}$ : là tọa độ y của điểm trái cùng của vật cản thứ i.

### 5.2.2.3 Tính danh sách khoảng trống và khoảng trống lớn nhất:



Hình 5-8 Danh sách khoảng trống

Độ lớn khoảng trống thứ nhất được xác định bằng góc biên quan sát bên trái trừ cho góc biên trái của vật cản đầu tiên:

$$gap[0] = \phi_{lim\_l} - \phi_{obs1\_l}$$

Các độ lớn khoảng trống tiếp theo bằng góc biên phải của vật cản thứ i trừ góc biên trái của vật cản thứ (i+1):

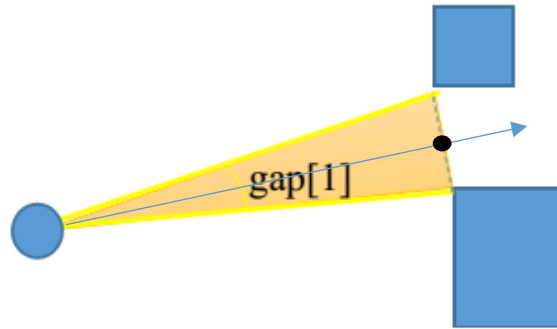
$$gap[i] = \phi_{obsi\_r} - \phi_{obs(i+1)_l}$$

Độ lớn khoảng trống cuối cùng bằng góc biên phải của vật cản cuối cùng trừ góc biên quan sát bên phải:

$$gap[n] = \phi_{obsn\_r} - \phi_{lim\_r}$$

Với n vật cản, sẽ có n + 1 khoảng trống là gap [0], ..., gap [i+1]. Từ đó ta tính được khoảng trống lớn nhất.

#### 5.2.2.4 Tính góc đến trung tâm khoảng trống lớn nhất:



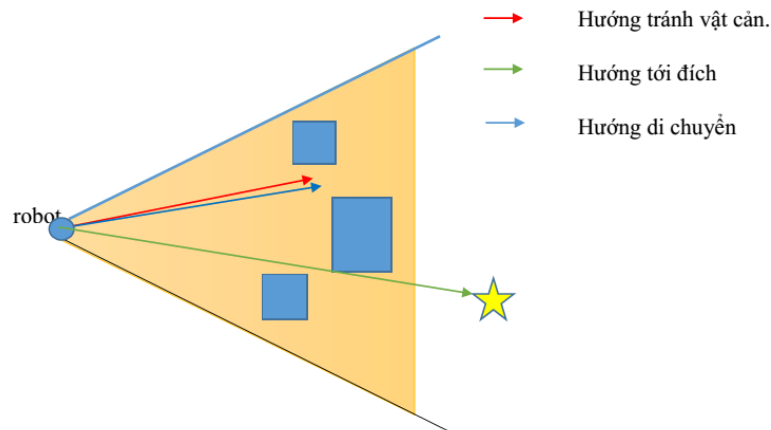
Hình 5-9 Đường trung tuyến của khoảng trống

Sau khi tìm được khoảng trống lớn nhất, ta sẽ chọn đường trung tuyến của góc khoảng trống, vì cách chọn đường trung tuyến giữ khoảng cách đều đối với hai vật cản bên đường đi của nó.

$$\phi_{\text{gap\_center}} = \phi_{\text{trung tuyến}}$$

$\phi_{\text{trung tuyến}}$  sẽ xác định bằng cách lấy điểm giữa của điểm bên trái cùng vật cản  $i$  và điểm bên phải cùng vật cản  $i+1$  rồi tính góc của điểm đó.

#### 5.2.2.5 Xác định hướng di chuyển cho robot:



Hình 5-10 Đường đi cho robot né tránh vật cản và hướng đến đích

Hướng đi cần tìm phải là hướng đi vừa né tránh vật cản vừa hướng tới đích, vì thế cần cân bằng việc hướng đích hoặc né tránh vật cản. Nếu ở xa vật cản, robot sẽ ưu tiên hướng đến đích và nếu gần vật cản, robot sẽ né tránh vật cản nhiều hơn hướng



đến đích. Từ kế hoạch trên, kết hợp hướng đến đích và hướng đến trung tâm khoảng cách lớn nhất như sau:

$$\phi_{final} = \frac{\left(\frac{\alpha}{d_{min}}\right)\phi_{gap\_center} + \phi_{goal}}{\left(\frac{\alpha}{d_{min}}\right) + 1} \quad (5.3)$$

Trong đó:

$\phi_{final}$ : hướng di chuyển theo phương pháp theo khoảng trống

$\alpha$ : hệ số điều khiển trọng số của hướng đích hoặc hướng né tránh

$d_{min}$ : khoảng cách ngắn nhất từ robot đến vật cản

$\phi_{gap\_center}$ : hướng đến tâm khoảng trống lớn nhất (né tránh vật cản)

$\phi_{goal}$ : hướng di chuyển thẳng đến đích

Trong công thức trên, nếu  $d_{min}$  lớn tức là robot xa vật cản thì trọng số  $\left(\frac{\alpha}{d_{min}}\right)$  sẽ bé khiến cho hướng đi đến tâm  $\phi_{gap\_center}$  chiếm tỷ lệ bé trong hướng đi cuối cùng  $\phi_{final}$  và robot chủ yếu đi theo hướng đến đích  $\phi_{goal}$ . Khi robot gần vật cản, tỷ số  $\left(\frac{\alpha}{d_{min}}\right)$  sẽ lớn khiến cho  $\phi_{gap\_center}$  chiếm tỷ lệ lớn và robot sẽ ưu tiên theo hướng né tránh vật cản, tuy nhiên cũng một phần hướng đích  $\phi_{goal}$  nhằm đảm bảo tính linh hoạt của hệ thống, không mãi theo hướng lớn nhất có thể dẫn dắt robot ngày càng xa vị trí đích.

## CHƯƠNG 6: THỰC NGHIỆM

*Dựa trên cơ sở lý thuyết đã trình bày ở các chương trước, ta có thể tiến hành thực nghiệm để đánh giá khả năng của phương pháp SLAM đã đề xuất. Chương này sẽ giới thiệu về chương trình mà khóa luận đã cài đặt cùng với những kết quả thu được. Ở đây khóa luận sẽ đánh giá trên một bộ dữ liệu từ camera stereo và IMU đã được công bố để so sánh các hệ thống SLAM.*

### 6.1 Tổng quan hệ thống

Chương 3 và 4 của khóa luận đã trình bày phần lý thuyết chủ yếu của một hệ thống SLAM sử dụng thông tin từ hai camera đồng bộ và IMU. Dựa vào cơ sở lý thuyết trên, khóa luận đã cài đặt được một chương trình hoàn chỉnh có khả năng đọc dữ liệu hình ảnh và IMU cho kết quả là vị trí của robot cùng với các vị trí 3D của đặc trưng thể hiện môi trường xung quanh. Chương trình có khả năng đọc dữ liệu đầu vào bằng cách:

- Dữ liệu đã được thu sẵn từ các bộ dữ liệu.

#### 6.1.1 Môi trường

Sau đây xin được trình bày cấu hình của máy tính mà khóa luận đã thử nghiệm hệ thống. Tất cả mọi kết quả thực nghiệm được trình bày trong đây đều được xử lý trên chính máy tính này.

- Hệ điều hành: Windows 10 Professional 64-bit
- Bộ xử lý: Intel Core i3 3110M 2.4GHz
- RAM: 4GB

#### 6.1.2 Thư viện hỗ trợ

Khóa luận đã cố gắng giảm số lượng thư viện cần sử dụng xuống mức tối thiểu. Các thư viện được sử dụng đều là các thư viện mã nguồn mở bao gồm:

- **OpenCV.** Thư viện này được dùng để cân chỉnh camera đồng thời rút trích và đối sánh các đặc trưng như từ các ảnh đầu vào. Ở đây khóa luận sử dụng OpenCV phiên bản 2.4.13.

### **6.1.3 Dữ liệu đầu ra:**

Đối với vị trí của robot, chương trình sẽ xuất ra dưới định dạng txt. Trong đó mỗi dòng của file kết quả sẽ ứng với một vị trí của robot.

## **6.2 Thực nghiệm trên bộ dữ liệu stereo của KITTY:**

### **6.2.1 Giới thiệu về bộ dữ liệu**

Bộ dữ liệu màu-độ sâu của Kitty là một bộ dữ liệu rất lớn bao gồm dữ liệu từ các cảm biến khác nhau như Stereo, IMU, laser, GPS cùng với dữ liệu đường đi chính xác với mục tiêu để so sánh, đánh giá các hệ thống SLAM. Bộ dữ liệu bao gồm nhiều đoạn phim khác nhau được thu bởi các camera. Mỗi đoạn phim này bao gồm một dãy ảnh màu từ hai camera và đầu ra của IMU, đường đi chính xác của cảm biến. Ở chương trình cài đặt, khóa luận sử dụng các đoạn phim được thu với tốc độ 10Hz và độ phân giải là 1242x375. Đường đi chính xác được lấy từ hệ thống GPS để đối sánh với kết quả.



Hình 6-1 Hình ảnh từ camera đồng bộ bộ dữ liệu KITTY, bên trên là ảnh từ camera chính, bên dưới là ảnh từ camera phụ cùng một thời điểm  
Đường link chung: [http://www.cvlibs.net/datasets/kitti/raw\\_data.php](http://www.cvlibs.net/datasets/kitti/raw_data.php)

Ở đây, khóa luận sử dụng các tập tin dữ liệu để thực nghiệm, vì lý do môi trường của các tập dữ liệu này là đường có nhà, cây cối, xe xung quanh, thuận lợi cho việc lấy đặc trưng. Đồng thời môi trường trong các tập dữ liệu này là động, tức là có những xe cộ, con người đang di chuyển nhằm thể hiện khả năng chống chịu ảnh hưởng từ môi trường động của khóa luận:

- 2011\_09\_26\_drive\_0005 (0.6 GB)

[http://kitti.is.tue.mpg.de/kitti/raw\\_data/2011\\_09\\_26\\_drive\\_0005/2011\\_09\\_26\\_drive\\_0005\\_sync.zip](http://kitti.is.tue.mpg.de/kitti/raw_data/2011_09_26_drive_0005/2011_09_26_drive_0005_sync.zip)

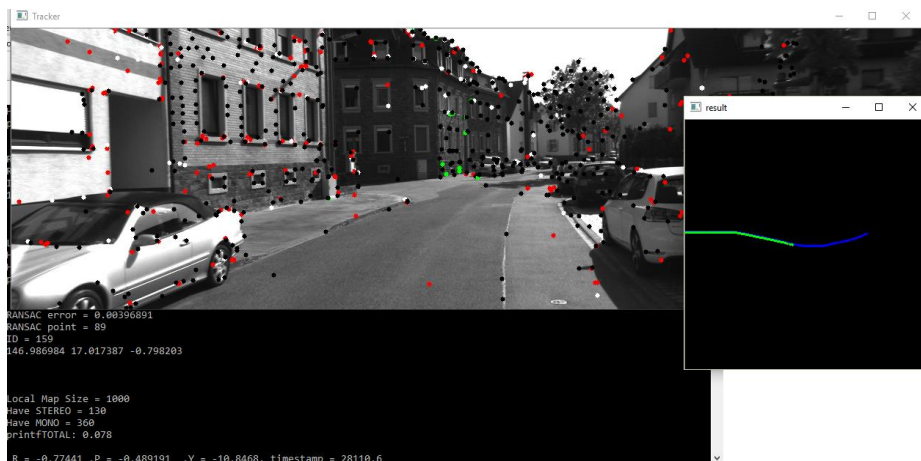
- 2011\_09\_26\_drive\_0048 (0.1 GB)

[http://kitti.is.tue.mpg.de/kitti/raw\\_data/2011\\_09\\_26\\_drive\\_0048/2011\\_09\\_26\\_drive\\_0048\\_sync.zip](http://kitti.is.tue.mpg.de/kitti/raw_data/2011_09_26_drive_0048/2011_09_26_drive_0048_sync.zip)

- 2011\_09\_26\_drive\_0060 (0.3 GB)  
[http://kitti.is.tue.mpg.de/kitti/raw\\_data/2011\\_09\\_26\\_drive\\_0060/2011\\_09\\_26\\_drive\\_0060\\_sync.zip](http://kitti.is.tue.mpg.de/kitti/raw_data/2011_09_26_drive_0060/2011_09_26_drive_0060_sync.zip)
- 2011\_09\_26\_drive\_0091 (1.3 GB)  
[http://kitti.is.tue.mpg.de/kitti/raw\\_data/2011\\_09\\_26\\_drive\\_0091/2011\\_09\\_26\\_drive\\_0091\\_sync.zip](http://kitti.is.tue.mpg.de/kitti/raw_data/2011_09_26_drive_0091/2011_09_26_drive_0091_sync.zip)
- 2011\_09\_26\_drive\_0095 (1.1 GB)  
[http://kitti.is.tue.mpg.de/kitti/raw\\_data/2011\\_09\\_26\\_drive\\_0095/2011\\_09\\_26\\_drive\\_0095\\_sync.zip](http://kitti.is.tue.mpg.de/kitti/raw_data/2011_09_26_drive_0095/2011_09_26_drive_0095_sync.zip)
- 2011\_09\_26\_drive\_0106 (0.9 GB)  
[http://kitti.is.tue.mpg.de/kitti/raw\\_data/2011\\_09\\_26\\_drive\\_0106/2011\\_09\\_26\\_drive\\_0106\\_sync.zip](http://kitti.is.tue.mpg.de/kitti/raw_data/2011_09_26_drive_0106/2011_09_26_drive_0106_sync.zip)
- 2011\_09\_26\_drive\_0113 (0.4 GB)  
[http://kitti.is.tue.mpg.de/kitti/raw\\_data/2011\\_09\\_26\\_drive\\_0113/2011\\_09\\_26\\_drive\\_0113\\_sync.zip](http://kitti.is.tue.mpg.de/kitti/raw_data/2011_09_26_drive_0113/2011_09_26_drive_0113_sync.zip)

Kết quả vị trí robot (xe) sẽ được so với dữ liệu vị trí từ GPS để cho ra cái nhìn tương đối về độ chính xác của thuật toán.

## 6.3 Kết quả thực nghiệm



Hình 6-2 Chạy chương trình trên bộ dữ liệu offline

Với phần ảnh thể hiện đoạn đường đi, các chấm là các đặc trưng theo vết theo thời gian. Phần result thể hiện hình chiếu 2D đường đi của xe với chiều ngang tương ứng trục x, chiều dọc là trục y, trục z không được thể hiện trên bản đồ vì độ cao của xe theo hệ tọa độ thế giới có giá trị thay đổi ít theo thời gian, đường màu xanh dương là tọa độ theo GPS, đường màu xanh lá là tọa độ tính theo thuật toán đề xuất.

$$\text{Ước lượng sai số: } MSE = \frac{1}{n} \sum_{t=1}^n (r_t - gps_t)^2 \quad (6.1)$$

Với: +  $r_t$  là vị trí robot thời điểm t ước lượng được.

+  $gps_t$  là vị trí GPS thời điểm t.

+ n : số thời điểm (tính theo số frame ảnh)

DATA	Time (s)	Số ảnh	Quãng đường (m)	Vận tốc (m/s)	MSE x	MSE y	MSE z
0005	15.4	154	61.6441	4	0.8181	0.5550	0.1281
0095	26.8	268	249.6923	9.3	0.2970	2.3032	3.0273
0113	8.8	88	17.43	2	0.0166	0.1633	0.1350
0060	7.8	78	0	0	0.0020	0.0020	0.0009
0091	33.9	339	196.0665	5.7	0.1381	0.2749	3.8126
0106	22.7	227	112.8423	4.97	0.0519	0.1604	0.2833

**Bảng 6-1 Kết quả chạy chương trình**

Có thể thấy kết quả chạy chương trình (hình 6-3) có độ chính xác trên trục x, y là tương đối cao. Tuy nhiên, với những bộ dữ liệu có đường đi là tương đối xa (0095 và 0091) sẽ gặp tình chung của hệ thống SLAM là sai số tích lũy và làm hệ thống

ngày càng lệch khỏi quỹ đạo đúng. Vì thế các hệ thống SLAM thường tích hợp thêm phát hiện vòng lặp để chỉnh lại các sai số tích lũy này khi quay về vị trí cũ.

Với bộ dữ liệu 0060, robot không di chuyển mà đứng im tại một chỗ. Trường hợp này với các hệ thống SLAM khác, vị trí thường bị trôi do việc xác định góc xoay bằng đặc trưng từng thời điểm. Nhưng với hệ thống đề xuất, kết quả cho thấy robot giữ vị trí rất tốt khi đứng im vì sử dụng trực tiếp góc xoay của IMU và áp dụng phương pháp lọc nhiễu đặc trưng RANSAC. Vì thế, hệ thống này có lợi thế hơn so với các hệ thống khác với các ứng dụng cần nhiều tác vụ robot giữ vững một vị trí (chụp ảnh trên cao, bay theo con người, ...)

# CHƯƠNG 7: KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN

*Khóa luận đã trình bày và tiến hành thực nghiệm một hệ thống SLAM sử dụng thiết bị camera Stereo và IMU làm cảm biến. Kết quả thực nghiệm cho thấy được thành công bước đầu của hệ thống mà khóa luận đã phát triển. Tuy nhiên, để có thể đưa đến một sản phẩm thực tế, còn nhiều vấn đề ta cần phải giải quyết. Chương này của khóa luận sẽ là để tổng kết và đưa ra các hướng phát triển trong tương lai.*

## 7.1 Tổng kết:

Bài toán về robot có khả năng tự hành vẫn luôn nhận được rất nhiều sự quan tâm từ cộng đồng khoa học máy tính. Tuy nhiên, ở Việt Nam thì lĩnh vực này nghiên cứu này vẫn còn khá mới mẻ và chưa có nhiều nghiên cứu đột phá. Hy vọng nghiên cứu này của khóa luận sẽ là bàn đạp cho những nghiên cứu sâu sắc hơn.

Khóa luận đề ra việc xây dựng được một hệ thống có khả năng định vị và tái tạo bản đồ ba chiều sử dụng dữ liệu đến từ hai camera và IMU có độ chính xác cao và khả năng xử lý trong thời gian thực. Những kết quả đã trình bày ở Chương 5 chính là minh chứng cho việc khóa luận đã hoàn thành mục tiêu đề ra này. Qua đó, khóa luận cũng đã giải quyết phần nào những thách thức còn tồn đọng hiện nay trong bài toán đó là nhiễu và tốc độ xử lý đồng thời mở rộng khả năng của SLAM ra môi trường ngoài trời và những môi trường không có GPS.

Các đóng góp chính của khóa luận như sau:

- Về mặt lý thuyết, khóa luận đã đưa ra một hệ thống có khả năng xử lý ở nhiều môi trường khác nhau, những môi trường không có tín hiệu từ hệ thống GPS đồng thời với việc kết hợp với tín hiệu IMU đã khiến hệ thống có khả năng đáp ứng được nhu cầu thực tế của robot như drone.



- Về mặt thực tiễn, khóa luận cũng đã cài đặt thành công một chương trình có khả năng ước lượng vị trí có khả năng hoạt động trong thời gian thực với đầu vào đến từ bộ dữ liệu được cho sẵn.

Tóm lại, đây là những kết quả bước đầu đáng khích lệ và mở ra nhiều hướng nghiên cứu mới trong tương lai nhằm hướng tới mục tiêu cuối cùng của lĩnh vực thị giác máy tính, làm sao để một tri thức nhân tạo có thể hiểu được ngữ nghĩa của thông tin thị giác như con người.

## **7.2 Hướng phát triển**

Vẫn còn nhiều khả năng với một hệ thống SLAM có thể hoạt động nhiều môi trường khác nhau và trong thời gian thực, việc mở rộng mô hình để đưa đến một sản phẩm thương mại cũng là một ý tưởng đáng để xem xét. Một số ứng dụng cụ thể có thể kể đến như hỗ trợ tìm đường cho người khiếm thị, drone cá nhân có khả năng bay theo và hỗ trợ với con người,... Ở đây xin được liệt kê một số hướng phát triển mà khóa luận có thể sử dụng trong tương lai.

### **7.2.1 Cải tiến tốc độ xử lý**

Tuy hệ thống đã phát triển có khả năng xử lý trong thời gian thực, nhưng việc tăng tốc độ xử lý luôn luôn được quan tâm đối với những ứng dụng cần phải biết vị trí càng chính xác càng tốt và cần tốc độ phản ứng nhanh như drone để đáp ứng nhu cầu thực tế. Một số cải tiến có thể được xem xét đó là lập trình song song hoặc đưa lên xử lý trên bộ xử lý đồ họa (GPU).

### **7.2.2 Phương pháp tối ưu toàn cục:**

Vì hệ thống chỉ tối ưu cục bộ với việc phát hiện và sửa lỗi đồng thời loại nhiễu. Tuy nhiên, khi robot di chuyển ngày càng nhiều, sai số sẽ tích lũy trong quá trình di chuyển và khiến hệ thống không thể đáp ứng được với những quãng đường dài. Chính vì thế, những phương pháp tối ưu toàn cục kết quả vị trí robot là cần được quan tâm. Một số phương pháp tối ưu toàn cục cho ra kết quả tốt như phương pháp tối ưu bằng đồ thị.

## DANH MỤC TÀI LIỆU THAM KHẢO

- [1] Arun , Somani K., Huang Thomas S., and Blostein Steven D., "Least-squares fitting of two 3-D point sets," in *Pattern Analysis and Machine Intelligence*, 1987, pp. 698-700.
- [2] Bachrach A., He R., and Roy N., "Autonomous flight in unknown indoor environments," in *Micro Air Vehicles*, 2009, pp. 217–228.
- [3] Bachrach A., Prentice S., He R., and Roy N., "RANGE-robust autonomous navigation in gps-denied environments," in *Robotics and Automation (ICRA)*, 2011, pp. 644–666.
- [4] Comport A., Malis E., and Rives P., "Accurate quadrifocal tracking for robust 3d visual odometry," in *Robotics and Automation*, 2007, pp. 40 –45.
- [5] Corke P. I., Strelow D., and Singh S., "Omnidirectional visual odometry for a planetary rover," in *Intelligent Robots and Systems*, 2005, pp. 4007 –4012.
- [6] Khoshelham , Kourosh , and Elberink Sander Oude, "Accuracy and resolution of kinect depth data for indoor mapping applications," , 2012, pp. 1437-1454.
- [7] Kummerle et al., "g2o: A general framework for graph optimization," in *Robotics and Automation*, 2011, pp. 3607-3613.
- [8] Lee , Sing Chee, Clark Daniel E., and Salvi Joaquim, "Slam with single cluster phd filters," in *Robotics and Automation (ICRA)*, 2012, pp. 543 - 552.
- [9] Lhuillier M., "Automatic structure and motion using a catadioptric camera," in *Workshop Omnidirectional Vision*, 2005, pp. 1-8.
- [10] Lowe and David G., "Distinctive image features from scale-invariant keypoints," *International journal of computer vision*, vol. 60, no. 2, pp. 91-110, 2004.
- [11] Mouragnon E., Lhuillier M., Dhome M., Dekeyser F., and Sayd P., "Real time localization and 3d reconstruction," in *Computer Vision and Pattern Recognition*, 2006, pp. 363 –370.

- [12] Nister D., "An efficient solution to the five-point relative pose problem," in *Computer Vision and Pattern Recognition*, 2003, pp. 195 –202.
- [13] Nister D., Naroditsky O., and Bergen J., "Visual odometry," in *Computer Vision and Pattern Recognition*, 2004, pp. 652 –659.
- [14] Olson E.B., "Real-time correlative scan matching," in *Robot. and Autom.*, 2009, pp. 4387–4393.
- [15] Olson E. B., "Robust and Efficient Robotic Mapping," MIT, Cambridge, MA, 2008.
- [16] Rusinkiewicz S. and Levoy M., "Efficient variants of the ICP algorithm," in *3-D Digital Imaging and Modeling*, 2001, pp. 145–152.
- [17] Sezer V. and Gokasan M., "A novel obstacle avoidance algorithm: ‘Follow the Gap Method’,” in *Elsevier - Robotics and Autonomous Systems*, 2012, pp. 1123–1134.
- [18] Shen S., Mulgaonkar Y., Michael Nathan, and Kumar V., "Vision-Based State Estimation and Trajectory Control Towards High-Speed Flight with a Quadrotor," in *Robotics: Science and Systems.*, 2013.
- [19] Shen S., Mulgaonkar Y., Michael Nathan, and Kumar V., "Vision-Based State Estimation for Autonomous Rotorcraft MAVs in Complex Environments," in *IEEE International Conference on Robotics and Automation*, 2013, pp. 1750–1756.
- [20] Tipaldi G. D. and Arras K. O., "FLIRT - interest regions for 2D range data," in *Robot. and Autom.*, 2010, pp. 3616–3622.