**ĐẠI HỌC BÁCH KHOA HÀ NỘI**

**ĐỒ ÁN TỐT NGHIỆP**

**Nghiên cứu phát triển một số giải pháp tăng độ chính xác cho hệ thống định vị toàn cầu**

**LÊ THÀNH LUÂN**

luan.lt203494@sis.hust.edu.vn

**Ngành Kỹ thuật điện tử viễn thống**

**Chuyên ngành Điện tử**

|  |  |
| --- | --- |
| **Giảng viên hướng dẫn:** | TS. Hoàng Phương Chi  Chữ ký của GVHD |
| **KHOA:** | Viễn thông |
| **HÀ NỘI, 6/2023** | |

**ĐỀ TÀI TỐT NGHIỆP**

Biểu mẫu của Đề tài/khóa luận tốt nghiệp/thực tập theo qui định của Trường, tuy nhiên cần đảm bảo giáo viên giao đề tài ký và ghi rõ họ và tên.

Trường hợp có 2 giáo viên hướng dẫn thì sẽ cùng ký tên.

Giáo viên hướng dẫn

Ký và ghi rõ họ tên

**Lời cảm ơn**

Đây là mục tùy chọn, nên viết phần cảm ơn ngắn gọn, tránh dùng các từ sáo rỗng, giới hạn trong khoảng 100-150 từ.

**Tóm tắt nội dung đồ án**

Đồ án này tập trung vào việc cải thiện độ chính xác của định vị GNSS bằng cách sử dụng bộ lọc Kalman sau bước ước lượng bình phương tối thiểu (WLS). Vấn đề cần thực hiện là nâng cao độ chính xác của vị trí và vận tốc trong hệ tọa độ ECEF từ dữ liệu GNSS. Phương pháp thực hiện bao gồm: sử dụng WLS để ước lượng ban đầu, sau đó áp dụng bộ lọc Kalman để tinh chỉnh các giá trị này. Ngoài ra bọn em còn sử dụng kỹ thuật Real Time Kinematic để làm kết quả chính xác nhất. Các công cụ sử dụng gồm ngôn ngữ lập trình Python và RTKLIB. Dữ liệu chúng em sử dụng từ tệp của Google từ cuộc thi GSDC2023. Kết quả của đồ án cho thấy bộ lọc Kalman giúp giảm thiểu sai số và cung cấp vị trí ước lượng chính xác hơn so với phương pháp WLS đơn thuần. Đồ án này có tính thực tế cao, có thể ứng dụng trong các hệ thống định vị và dẫn đường. Định hướng phát triển mở rộng có thể bao gồm việc tích hợp thêm các cảm biến khác như IMU để cải thiện thêm độ chính xác. Qua đồ án, em đã nắm vững kiến thức về lý thuyết và ứng dụng của WLS, bộ lọc Kalman, lý thuyết xác suất thống kê, ma trận, kỹ năng lập trình với Python, cùng với khả năng phân tích và xử lý dữ liệu GNSS.

Sinh viên thực hiện

Ký và ghi rõ họ tên

**MỤC LỤC**

[CHƯƠNG 1. GIỚI THIỆU 1](#_Toc171086880)

[1.1 Lý do chọn đề tài 1](#_Toc171086881)

[1.2 Mục tiêu nghiên cứu 1](#_Toc171086882)

[1.3 Tổng quan các phần 1](#_Toc171086883)

[CHƯƠNG 2. CƠ SỞ LÝ THUYẾT 2](#_Toc171086884)

[2.1 Tổng quan Kalman Filter 2](#_Toc171086885)

[2.2 Cấu trúc bộ lọc Kalman 2](#_Toc171086886)

[2.2.1 Bước dự đoán ( Prediction step): 3](#_Toc171086887)

[2.2.2 Bước cập nhật (Update step): 5](#_Toc171086888)

[2.2.3 Tóm tắt 6](#_Toc171086889)

[2.3 Khoảng cách Mahalanobis 8](#_Toc171086890)

[CHƯƠNG 3. THIẾT KẾ VÀ TRIỂN KHAI HỆ THỐNG 8](#_Toc171086891)

[3.1 Thiết kế hệ thống 8](#_Toc171086892)

[3.1.1 Đặc tả Kalman Filter 9](#_Toc171086893)

[3.1.2 Sơ đồ luồng dữ liệu 11](#_Toc171086894)

[3.2 Triển khai hệ thống 12](#_Toc171086895)

[3.2.1 Sơ đồ thuật toán 12](#_Toc171086896)

[CHƯƠNG 4. TRIỂN KHAI THỰC NGHIỆM 14](#_Toc171086897)

[4.1 Triển khai thực nghiệm 14](#_Toc171086898)

[4.1.1 Mô tả thực nghiệm 14](#_Toc171086899)

[4.1.2 Phương pháp đánh giá thực nghiệm 14](#_Toc171086900)

[4.2 Kết quả thực nghiệm 17](#_Toc171086901)

[CHƯƠNG 5. KẾT LUẬN 21](#_Toc171086902)

[TÀI LIỆU THAM KHẢO 22](#_Toc171086903)

[PHỤ LỤC 23](#_Toc171086904)

**DANH MỤC HÌNH VẼ**

[Hình 2.1 Mô tả dự đoán và cập nhật điển hình của bộ lọc Kalman 6](#_Toc170769439)

[Hình 2.2 Sơ đồ hoàn chỉnh về hoạt động của Bộ lọc Kalman. 6](#_Toc170769440)

**DANH MỤC HÌNH VẼ**

**No table of figures entries found.**

# GIỚI THIỆU

## Lý do chọn đề tài

## Mục tiêu nghiên cứu

## Tổng quan các phần

Chương 1. Giới thiệu

Chương 2. Cơ sở lý thuyết

Chương 3. Phương pháp nghiên cứu và triển khai

Chương 4. Kết quả và thảo luận

Chương 5. Kết luận và hướng phát triển

Tài liệu tham khảo

Phụ lục

# CƠ SỞ LÝ THUYẾT

## Tổng quan Kalman Filter

Hầu hết các hệ thống hiện đại được trang bị nhiều cảm biến cung cấp ước tính các biến ẩn (chưa biết) dựa trên một loạt các phép đo. Ví dụ: máy thu GPS cung cấp ước tính vị trí và vận tốc, trong đó vị trí và vận tốc là các biến ẩn và chênh lệch thời gian của tín hiệu đến của các vệ tinh là phép đo.

Một trong những thách thức lớn nhất của hệ thống theo dõi và kiểm soát là cung cấp đúng và chính xác các dự đoán của biến ẩn khi không chắc chắn dữ liệu. Trong máy thu GPS, độ không đảm bảo của máy phụ thuộc vào nhiều yếu tố bên ngoài như nhiễu nhiệt, hiệu ứng khí quyển, sự thay đổi nhỏ của vị trí vệ tinh, độ chính xác của đồng hồ máy thu và nhiều yếu tố khác.

Kalman Filter là một trong những thuật toán ước lượng quan trọng và phổ biến nhất. Bộ lọc Kalman tạo ra các ước tính về các biến ẩn dựa trên các phép đo không chính xác và không chắc chắn. Ngoài ra, Kalman Filter còn cung cấp dự đoán về tương lai của trạng thái của hệ thống dựa trên các dự báo trong quá khứ.

Bộ lọc này được đặt tên theo Rudolf E. Kálmán (19 tháng 5, 1930 – 2 tháng 7, 2016). Năm 1960, Kálmán xuất bản bài báo nổi tiếng của mình mô tả một giải pháp đệ quy (recursive) cho vấn đề lọc dữ liệu tuyến tính rời rạc.

Ngày nay, Kalman Filter được sử dụng trong theo dõi mục tiêu (Radar), hệ thống định vị, hệ thống điều khiển, đồ họa máy tính và nhiều hơn nữa.

Bộ lọc Kalman ( Kalman Filter) có nhiều loại như: LKF (Linear Kalman Filter), Non-LKF (Non-Linear Kalman Filter), EKF (Extended Kalman Filter), ... Nhưng trong đồ án này bộ lọc Kalman tuyến tính(LKF) được lựa chọn làm giải pháp cho ứng dụng định vị vì có hiệu suất tối ưu về sai số ước tính bình phương trung bình tối thiểu (theo một số giả định).

## Cấu trúc bộ lọc Kalman

*Kí hiệu được sử dụng chung:*

* Chữ thường, in đậm đề cập tới vector:
* Chữ in hoa, in đậm đề cập tới ma trận:
* Chữ thường, không in đậm là phần tử trong vector:
* Chữ in hoa, không in đậm là những phần tử trong ma trận:

Cấu trúc của một bộ lọc Kalman gồm có 2 phần tham khảo [1] [2] mô tả các bước dự đoán và cập nhật.

A diagram of a function

Description automatically generated

Hình 2.1 Mô tả dự đoán và cập nhật điển hình của bộ lọc Kalman

Kí hiệu : dùng để diễn tả trạng thái ước tính

* **Bước dự đoán (chỉ số là dấu -)** : Từ trạng thái ước tính tại thời điểm n−1, sử dụng ma trận chuyển đổi trạng thái **F**, ma trận điều khiển **G**, và tín hiệu điều khiển để ước tính trạng thái tiếp theo tại thời điểm n.
* **Bước cập nhật(chỉ số là dấu +)**: Khi nhận được giá trị đo lường tại thời điểm n, trạng thái ước tính được điều chỉnh để đưa ra trạng thái chính xác hơn .

Quá trình này lặp lại cho các thời điểm tiếp theo n+1 với các bước tương tự, từ đến rồi , duy trì tính chính xác của trạng thái hệ thống qua các bước dự đoán và cập nhật liên tục.

### Bước dự đoán ( Prediction step):

#### Phương trình ngoại suy trạng thái

|  |  |  |
| --- | --- | --- |
|  |  | PT. 2.1 |

|  |  |
| --- | --- |
| Trong đó: |  |
| : | Vecto trạng thái ước tính của bước dự đoán tại thời điểm n |
| : | Vecto trạng thái ước tính của bước cập nhật tại thời điểm n - 1 |
| : | Vecto đầu vào điều khiển |
| : | Ma trận chuyển trạng thái |
| : | Ma trận điều khiển |

Ước tính trạng thái hệ thống x tại thời điểm n sử dụng tại thời điểm n - 1 được gọi là dự đoán kết hợp với tích đầu vào điều khiển. Nó cũng là ước tính tiên nghiệm vì nó áp dụng *trước* cho một phép đo.

Trong đồ án này phương trình được triển khai từ một mô hình tuyến tính chuyển động có mối quan hệ giữa vị trí(khoảng cách) và vận tốc của hệ chất điểm [3]:

Từ đó có thể rút ra các thành phần tương ứng sau theo lý thuyết ma trận [4] và phương trình PT. 2.10:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  |

*Chú ý:* trong báo cáo này = 1: khoảng thời gian để đo được hay lấy mẫu có giá trị bằng 1

#### Phương trình ngoại suy hiệp phương sai

|  |  |  |
| --- | --- | --- |
|  |  | PT. 2.2 |

|  |  |
| --- | --- |
| Trong đó: |  |
| : | Ma trận hiệp phương sai của trạng thái ước tính tại thời điểm |
| : | Ma trận hiệp phương sai của trạng thái ước tính tại thời điểm (dùng để ước tính cho thời điểm ) |
| : | Ma trận chuyển trạng thái |
| : | Ma trận hiệp phương sai nhiễu quá trình |
| : | Chuyển vị của ma trận chuyển trạng thái |

Bộ lọc Kalman tính toán và truyền đi hiệp phương sai lỗi trạng thái tại thời điểm n được cung cấp bởi thông tin của thời điểm n-1. Nó được biểu diễn bằng ma trận hiệp phương sai , cũng được biết đến như là ma trận hiệp phương sai tiên nghiệm, là thước đo mức độ chính xác của các ước tính so với hệ thống thực. biểu diễn cho ma trận hiệp phương sai ước tính tốt nhất ở thời điểm trước n-1 (+).

Ở trong báo cáo này phục thuộc vào sẽ là ma trận hiệp phươngsai của vận tốc,là ma trận hiệp phương sai củacủa trạng thái ước tính tại thời điểm n-1 (+) và ma trận như phần 2.2.1.1

### Bước cập nhật (Update step):

#### Phương trình phần dư đo lường

Phần dư đo lường cho biết các phép đo khớp với mô hình đo lường tốt như thế nào, tức là các phép đo thực tế khác với các phép đo dự đoán như thế nào.

|  |  |  |
| --- | --- | --- |
|  |  | PT. 2.3 |

|  |  |  |  |
| --- | --- | --- | --- |
| Trong đó: | | |  |
| **:** | | | Veco nhiễu ngẫu nhiên(phần dư đo lường) tại thời điểm n |
| **:** | | | Vecto đo lường chứa giá trị đo lường thực tế tại thời điểm n |
| **:** | | | Ma trận đo lường |
| : | Vecto trạng thái được ước tính tại thời điểm n(-) của bước dự đoán | | |

Ma trận đo lường ở đây có thể dùng để làm tỉ lệ (scaling), lựa chọn trạng thái (state selection) và kết hợp các trạng thái (combination of states). Ở đây vì sử dụng để quan sát các giá trị x,y,z nên ma trận sẽ đóng vai trò lựa chọn trạng thái như ma trận đơn vị 3x3 (xem lại ma trận đơn vị ở đây [4]).

là một dấu hiệu cho thấy mô hình dự đoán so với các giá trị đo lường thực tế tốt như thế nào. Nếu sự khác biệt này bằng 0, điều đó có nghĩa là các phép đo thực tế bằng với các giá trị đã dự đoán, do đó ước tính tiên nghiệm được giả định là ước tính tốt nhất của hệ thống thực.

#### Phương trình độ lợi Kalman – trọng số (Kalman Gain):

|  |  |  |
| --- | --- | --- |
|  |  | PT. 2.4 |

|  |  |
| --- | --- |
| Trong đó: |  |
| : | Độ lợi Kalman (Kalman Gain) |
| : | Ma trận hiệp phương sai ở phần dự đoán |
| : | Ma trận đo lường |
| : | Ma trận hiệp phương sai nhiễu đo lường |
| : | Độ lợi Kalman (Kalman Gain) |

Độ lợi Kalman chỉ đơn giản là cân nhắc tác động từ mô hình so với các phép đo. Khi càng nhỏ và càng nhỏ thì nhỏ , ngược lại sẽ rất lớn khi rất nhỏ. được sử dụng ở trong báo cáo này sẽ là ma trận hiệp phương sai của giá trị đo lường thực tế tại thời điểm n.

#### Phương trình cập nhật trạng thái

|  |  |  |
| --- | --- | --- |
|  |  | PT. 2.5 |

|  |  |
| --- | --- |
| Trong đó: |  |
| : | Vecto trạng thái ước tính(đầu ra hệ thống) |
| : | Vecto trạng thái ước tính ở phần dự đoán |
| : | Độ lợi Kalman (Kalman Gain) |
| **:** | Vecto nhiễu ngẫu nhiên(phần dư đo lường) |

Nếu độ lợi bằng không, điều đó có nghĩa là chỉ có mô hình được tin cậy và khi độ lợi được tăng lên, nó có thể được coi là sự tin tưởng hơn vào các phép đo .

#### Phương trình ma trận cập nhật hiệp phương sai

|  |  |  |
| --- | --- | --- |
|  |  | PT. 2.6 |

|  |  |
| --- | --- |
| Trong đó: |  |
| : | Ma trận hiệp phương sai ước tính của trạng thái hiện tại n(+). |
| : | Ma trận hiệp phương sai ước tính ở phần dự đoán |
| **:** | Ma trận đo lường |
| **:** | Ma trận đơn vị |
| : | Độ lợi Kalman (Kalman Gain) |

Dựa trên giá trị của , bộ lọc Kalman cũng cập nhật sự không chắc chắn của dự đoán mới của nó . Ở báo cáo này ma trận là ma trận đơn vị 3x3

### Tóm tắt

Các phương trình dự đoán và cập nhật của thuật toán Kalman Filter sẽ được trình bày tóm tắt Hình 2.2 và các ký hiệu ở Bảng 2.1 .

A diagram of mathematical equations

Description automatically generated

Hình 2.2 Sơ đồ hoàn chỉnh về hoạt động của Bộ lọc Kalman.

|  |  |
| --- | --- |
| ***Kí hiệu*** | ***Ý nghĩa*** |
|  | Vecto trạng thái được ước tính tại thời điểm n của bước dự đoán |
|  | Vecto trạng thái được ước tính tại thời điểm n-1 của bước cập nhật |
|  | Vecto đầu vào điều khiển dùng để ước tính trạng thái tại bước dự đoán. |
|  | Ma trận chuyển trạng thái |
|  | Ma trận điều khiển |
|  | Ma trận hiệp phương sai tại thời điểm n-1 của bước cập nhật |
|  | Ma trận hiệp phương sai tại thời điểm n của bước dự đoán |
|  | Ma trận hiệp phương sai nhiễu quá trình |
|  | Vecto trạng thái ước tính(đầu ra hệ thống) |
|  | Vecto đo lường tại thời điểm n |
|  | Vecto phần dư đo lường |
|  | Ma trận đo lường |
|  | Ma trận đơn vị |
|  | Độ lợi Kalman (Kalman Gain) |
|  | Ma trận hiệp phương sai nhiễu đo lường ứng với |

Bảng 2.1 Bảng tóm tắt kí hiệu

## Khoảng cách Mahalanobis

Công thức được tham khảo từ [5]:

|  |  |  |
| --- | --- | --- |
|  |  | PT. 2.7 |

Tính khoảng cách từ một điểm đến tâm (μ) được xác định bởi các thuộc tính liên quan với ma trận hiệp phương sai (C), là ma trận ứng với .

Ở báo cáo này các tọa độ x,y,z theo hệ tọa độ ECEF sẽ được sử dụng tính toán khoảng cách Mahalanobis để loại bỏ các điểm bất thường. Khoảng cách Mahalanobis được áp dụng cho công thức PT. 2.12 trong bước cập nhật và ma trận P công thức PT. 2.11 trong bước dự đoán của Kalman Filter. Ta có công thức khoảng cách Mahalanobis với từng lần n như sau:

|  |  |  |
| --- | --- | --- |
|  |  | PT. 2.8 |

|  |  |
| --- | --- |
| Trong đó: |  |
| : | Khoảng cách Mahalanobis tại thời điểm n |
| : | Phần dư đo lường PT. 2.12 | |
| : | Chuyển vị của phần dư đo lường |
| : | Ma trận hiệp phương sai của tại mỗi thời điểm n |
|  | |

# THIẾT KẾ VÀ TRIỂN KHAI HỆ THỐNG

## Thiết kế hệ thống

### Đặc tả Kalman Filter

A diagram of a filter

Description automatically generated

Hình 3.1 Sơ đồ khối tổng quan Kalman Filter

Đầu vào của khối Kalman Filter được lấy từ đầu ra của khối WLS trước đó gồm:

* **x\_wls:** mảng 1 chiều với 3 phần tử để lưu tọa độ (x,y,z) .
* **v\_wls:** mảng 1 chiều với 3 phần tử để lưu vận tốc của tọa độ (x,y,z).
* **cov\_x:** mảng 2 chiều 3x3 để lưu ma trận hiệp phương sai tọa độ.
* **cov\_v:** mảng 2 chiều 3x3 để lưu ma trận hiệp phương sai vận tốc.

Đầu ra của khối Kalman Filter:

* **x\_kf:** mảng 1 chiều với 3 phần tử để lưu tọa độ (x,y,z)

Trong khối chức năng Kalman Filter bao gồm các khối nhỏ hơn:

A diagram of a flowchart

Description automatically generated

Hình 3.2 Các khối chính trong Kalman Filter

Mô tả ngắn gọn Hình 3.2 :

* **Khởi tạo:**

Các biến để lưu trữ các giá trị của tọa độ, vận tốc và các ma trận đơn vị, ma trận hiệp phương sai.

* **Predict (Bước dự đoán)**:

Dự đoán các giá trị tọa độ tiếp theo PT. 2.1 bằng giá trị tọa độ và vận tốc trung bình hiện tại. Ngoài ra còn sự phân tán, sai lệch về các tọa độ còn được dự đoán bằng ma trận hiệp phương sai PT. 2.2.

* **Kiểm tra khoảng cách Mahalanobis:**

Kiểm tra khoảng cách Mahalanobis bằng cách thực hiện tính toán với các giá trị tọa độ, ma trận hiệp phương sai được dự đoán từ Predict (bước dự đoán) và tọa độ đo được để loại bỏ các giá trị tọa độ ngoại lai.

* **Update (Bước cập nhật):**

Cập nhật các giá trị tọa độ đã dự đoán từ trước bằng các PT. 2.3, PT. 2.4, PT. 2.5, PT. 2.6 để đưa ra được tọa độ chuẩn xác hơn.

### Sơ đồ luồng dữ liệu

Các kí hiệu trong sơ đồ ứng với biểu thức toán học về Kalman Filter được biểu diễn rõ trong Bảng 2.1. Ở đây các chỉ số được nhắc lại với chỉ số n là tại thời điểm n, chỉ số n(-) là dự đoán tại thời điểm n và chỉ số n(+) là cập nhật tại thời điểm n. Ý nghĩa của sơ đồ được trình bày dưới hình ảnh.

**A diagram of a process

Description automatically generated**

Hình 3.3 Sơ đồ luồng dữ liệu ước tính tọa độ của Kalman Filter

* Tọa độ WLS thực tế đo được đầu tiên đưa vào xử lí trong khối Predict(Bước dự đoán) ứng với và ma trận hiệp phương sai ngoại suy gồm các biểu thức trình bày phần 2.2.1.
* Dữ liệu đi ra là tọa độ và ma trận hiệp phương sai dự đoán , .
* Tiếp theo dữ liệu được đưa vào khối kiểm tra khoảng cách Mahalanobis sử dụng các biểu thức phần 2.3.
  + Nếu kiểm tra khoảng cách sai, nghĩa là lớn hơn ngưỡng được đưa ra thì điều chỉnh ma trận hiệp phương sai dự đoán , tọa độ dự đoán được giữ nguyên và đưa ra tọa độ và ma trận hiệp phương sai cập nhật cho trạng thái tiếp theo và .
  + Còn nếu kiểm tra khoảng cách Mahalanobis đúng, nghĩa là nhỏ hơn ngưỡng thì thực hiện bước Update(Bước cập nhật) gồm các biểu thức phần 2.2.2, dữ liệu từ ma trận là hiệp phương sai tọa độ đo thực tế từ WLS kết hợp với , đã giải thích từ trước. Từ đó cho ra được dữ liệu cập nhật và tốt cho thời điểm n.
* Sau đó và với lần lặp tiếp theo tương ứng với và được đưa trở lại khối Predict và thực hiện các thao tác tương tự.

## Triển khai hệ thống

### Sơ đồ thuật toán

Sơ đồ thuật toán hàm kalman\_filter được mô tả bằng hình vẽ như sau:

A diagram of a algorithm

Description automatically generated

Hình 3.4 Sơ đồ thuật toán Kalman Filter

**1. Khởi tạo**:

* F: ma trận chuyển trạng thái là ma trận đơn vị 3x3 dưới dạng một mảng 2 chiều.
* H: ma trận quan sát là ma trận đơn vị 3x3 dưới dạng một mảng 2 chiều.
* P: ma trận hiệp phương sai cập nhật 3x3 dưới dạng một mảng 2 chiều.
* I: ma trận đơn vị 3x3 dưới dạng một mảng 2 chiều.
* x\_pred: mảng 1 chiều với 3 phần tử x,y,z dự đoán
* P\_pred: ma trận hiệp phương sai dự đoán dưới dạng mảng 2 chiều.
* x\_kf: mảng 1 chiều với 3 phần tử x,y,z lưu lại giá trị.
* P\_kf: ma trận hiệp phương sai cập nhật dưới dạng mảng 2 chiều.
* sigma\_mahalanobis: là một biến số với 1 giá trị ngưỡng cố định.

**2. vs = v\_wls trung bình hai điểm**:

* vs: giá trị vận tốc trung bình giữa điểm được ước tính và điểm trước đó, mục đích sử dụng giá trị vận tốc trung bình để tăng độ chính xác cho bước dự đoán.
* vs: là mảng 1 chiều

**3. index = min(len(vs), len(x\_wls))**:

* Xác định chỉ số nhỏ hơn giữa chiều dài của **vs** và **x\_wls** nghĩa là xác định số lượng các số lượng vận tốc vs và tọa độ (x,y,z).

**4. i = 0:**

* Bắt đầu vòng lặp với chỉ số ban đầu bằng 0 thì chuyển sang vòng lặp tiếp theo.

**5. i++**:

* Tăng giá trị i lên 1 để thực hiện vòng lặp tiếp theo.

**6. x\_pred, P\_pred = predict(x, u, P, Q)**:

* Thực hiện Predict (Bước dự đoán) dựa vào x, u, P và Q để dự đoán các giá trị x\_pred và P\_pred.

**7. Kiểm tra khoảng cách Mahalanobis**:

* **d = mahalanobis(H.x\_pred, z, inv(P\_pred))**: Tính khoảng cách Mahalanobis PT. 2.8.
* **d < sigma\_mahalanobis?**: Kiểm tra xem **d** có nhỏ hơn ngưỡng **sigma\_mahalanobis** không.

**8. x, P = update(x\_pred, P\_pred, z, R)**:

* Nếu d < sigma\_mahalanobis là đúng, thực hiện Update (Bước cập nhật) phần 2.2.2 ,dựa vào x\_pred, P\_pred, z, R để cập nhật các giá trị x\_pred và P\_pred.

**9. P\_pred = P \* 100 + Q:**

* Nếu d < sigma\_mahalanobis là sai, thực hiện cập nhật lại P\_pred.

**10. x\_kf[i] = x**:

* Lưu lại giá trị tọa độ ước tính chính xác sau mỗi lần lặp vào mảng x\_kf[i].

**11. i index?**:

* Kiểm tra xem chỉ số hiện tại có bằng **index** không.
  + Nếu đúng, trả về **x\_kf[index]** và kết thúc hàm.
  + Nếu sai, thực hiện lại từ bước **5.**

# TRIỂN KHAI THỰC NGHIỆM

## Triển khai thực nghiệm

### Mô tả thực nghiệm

Theo dữ liệu được cung cấp bới Google, độ chính xác vị trí của điện thoại nằm trong khoảng từ 3-5m. Để xác lập được kết quả cho các tọa độ của điện thoại thì trước hết chúng ta cần các điểm chuẩn (ground truth) cùng thời gian mà điện thoại lấy mẫu. Các điểm ground truth sẽ có độ chính xác cao nhất có thể. Hệ thống NovAtel SPAN và Waypoint Inertial Explorer được sử dụng để tạo ra kết quả chuẩn, thông qua việc kết hợp chặt chẽ các phép đo IMU và GNSS. Để loại bỏ các sai lệch do các hệ tọa độ khác nhau và chuyển động kiến tạo, toàn bộ dữ liệu chuẩn được tạo ra trong hệ tọa độ WGS84 với tọa độ của tất cả các trạm gốc.

A diagram of a vehicle

Description automatically generated

Hình 4.1 Cách thức Google lấy dữ liệu chuẩn

### Phương pháp đánh giá thực nghiệm

#### Tính khoảng cách giữa hai tọa độ

Để xác định sai số của một tọa độ chúng em sẽ tiến hành so sánh khoảng cách giữa điểm ấy và ground truth trong hệ WGS84 tại cùng thời gian. Để tính được khoảng cách này chúng em sẽ sử dụng công thức **Vincenty**. Công thức của Vincenty là các phương pháp được sử dụng trong địa lý để tính khoảng cách giữa hai điểm trên bề mặt của mô hình Trái Đất hình elipse. Phương pháp này chính xác hơn các phương pháp giả định Trái Đất là một hình cầu, chẳng hạn như khoảng cách theo đường tròn lớn. Các công thức của **Vincenty** có độ chính xác trong vòng 1 mm hoặc ít hơn trên elipse tham chiếu WGS84.

Bảng 4.1 Bảng mô tả ký hiệu của công thức tính khoảng cách Vincenty

|  |  |  |
| --- | --- | --- |
| **Ký hiệu** | **Định nghĩa** | **Giá trị** |
| *a* | Độ dài bán trục lớn của Trái Đất | 6378137.0m |
| *ƒ* | Độ dẹt của ellipsoid | 1/298.257223563 |
| *b* = (1 − *ƒ*) *a* | Độ dài bán trục bé Trái Đất | 6356752.3142m |
| *Φ*1, *Φ*2 | Vĩ độ của các điểm |  |
| *U*1 = arctan( (1 − *ƒ*) tan *Φ*1 ),  *U*2 = arctan( (1 − *ƒ*) tan *Φ*2 ) | Vĩ độ trên mặt phẳng auxiliary |  |
| *L*1, *L*2 | Kinh độ của các điểm |  |
| *L* = *L*2 − *L*1 | Độ lệch giữa 2 giá trị kinh độ |  |
| *λ* | Độ lệch giữa 2 giá trị kinh độ trên mặt phẳng auxiliary |  |
| *α*1, *α*2 | Góc phương vị tại các điểm |  |
| *α* | Góc phương vị tại đường xích đạo |  |
| *s* | Khoảng cách giữa 2 điểm |  |
| *σ* | Khoảng cách góc giữa 2 điểm |  |
| *σ*1 | Khoảng cách góc giữa 1 điểm và xích đạo |  |
| *σ*m | Khoảng cách góc của điểm nằm giữa và xích đạo |  |

Dựa vào tọa độ của hai điểm (Φ1, L1) và (Φ2, L2) chúng em sẽ đi tìm khoảng cách s.

Đầu tiên ta cần tính U1, U2 và L, và đặt giá trị ban đầu của λ = L. Sau đó, lặp lại việc đánh giá các phương trình sau cho đến khi λ hội tụ:

Khi hội tụ đến giá trị chính xác (0.06mm) thì tiếp tục các phương trình:

Công thức để tính khoảng cách giữa 2 điểm:

|  |  |  |
| --- | --- | --- |
|  |  | PT. 4.1 |

#### Tính kết quả cho một tập dữ liệu

Để tính giá trị sai số cho một tập điểm chúng em dựa trên công thức mà Google cung cấp: **trung bình** của các khoảng cách phân vị thứ 50 và 95. Các khoảng cách này tạo thành một phân phối từ đó khoảng cách phân vị thứ 50 (50% các khoảng cách nhỏ hơn giá trị này) và 95 (95% các khoảng cách nhỏ hơn giá trị này) được tính toán:

|  |  |  |
| --- | --- | --- |
|  |  | PT. 4.2 |

* d là kết quả đạt được của tập dữ liệu
* s là tập khoảng cách của từng điểm so với giá trị chuẩn của nó
* quantile(s,50) là phân vị 50 của tập s
* quantile(s,95) là phân vị 95 của tập s

A diagram of a blue circle with red and green text

Description automatically generated

Hình 4.1 Hình mô tả phân vị 50 và 95

## Kết quả thực nghiệm

Ở phần này kết quả sẽ được trình bày bằng biểu đồ sai số khoảng cách giữa hai điểm cùng thời gian của Baseline và KF qua 3 tệp dữ liệu sau:

2020-06-25-00-34-us-ca-mtv-sb-101/pixel4xl

2021-07-19-20-49-us-ca-mtv-a/sm-g988b

2023-09-07-19-33-us-ca/pixel5a

Sau đó số điểm của các tệp sẽ được thống kê và so sánh ở biểu đồ giá trị lỗi khoảng cách với:

* Trục tung: thể hiện các giá trị sai số khoảng cách giữa Baseline và Kalman Filter.
* Trục hoành: thể hiện số lượng mẫu được so sánh cùng thời điểm.

#### 2020-06-25-00-34-us-ca-mtv-sb-101/pixel4xl

A graph with blue and orange lines

Description automatically generated

Hình 4.2 Các sai số khoảng cách giữa Baseline và Kalman Filter

Biểu đồ thể hiện sai số khoảng cách cho thiết bị Pixel 4XL vào ngày 25 tháng 6 năm 2020. Đường biểu diễn màu xanh dương đại diện cho Baseline với điểm số trung bình là 2.0640 mét, trong khi đường màu cam đại diện cho phương pháp Kalman Filter với điểm số trung bình là 1.5806 mét. Kalman Filter giúp giảm đáng kể sai số so với Baseline.

#### 2021-07-19-20-49-us-ca-mtv-a/sm-g988b

A graph with blue lines and orange lines

Description automatically generated

Hình 4.3 Các sai số khoảng cách giữa Baseline và Kalman Filter

Biểu đồ thể hiện sai số khoảng cách cho thiết bị SM-G988B vào ngày 19 tháng 7 năm 2021. Điểm số trung bình của Baseline là 3.9466 mét, và của phương pháp Kalman Filter là 3.5692 mét. Kalman Filter một lần nữa cho thấy cải thiện độ chính xác, mặc dù mức giảm không quá lớn so với Baseline.

#### 2023-09-07-19-33-us-ca/pixel5a

A graph with blue lines and orange lines

Description automatically generated

Hình 4.4 Các sai số khoảng cách giữa Baseline và Kalman Filter

Biểu đồ thể hiện sai số khoảng cách cho thiết bị Pixel 5a vào ngày 7 tháng 9 năm 2023. Baseline có điểm số trung bình là 5.7372 mét, trong khi phương pháp Kalman Filter có điểm số trung bình là 3.5532 mét. Kalman Filter giảm đáng kể sai số, cho thấy khả năng cải thiện đáng kể độ chính xác vị trí.

|  |  |  |
| --- | --- | --- |
| **Độ chính xác**    **Điện thoại** | **Kết quả nghiên cứu**  **(Khối Kalman Filter)** | **Google baseline** |
| **Pixel4xl** | 1.5806m | 2.6040m |
| **Pixel5a** | 3.5532m | 5.7372m |
| **Samsung Galaxy S21 Ultra** | 3.5692m | 3.9466m |

Bảng 4.1 Bảng đánh giá kết quả Baseline và Kalman Filter

# KẾT LUẬN

# TÀI LIỆU THAM KHẢO

|  |  |
| --- | --- |
| [1] | A. Becker, "Kalman FIlter," [Online]. Available: https://www.kalmanfilter.net/VI/background\_vi.html. |
| [2] | T. B. K. •. J. G. Aboelmagd Noureldin, Fundamentals of Inertial Navigation, Satellite-based Positioning and their Integration. |
| [3] | L. D. B. v. c. t. g. khác, Giáo trình Vật lý đại cương – 3 tập Lý thuyết và Bài tập, NXB Giáo dục.. |
| [4] | S. Hust, "Algebra, MI1141," [Online]. Available: https://fami.hust.edu.vn/wp-content/uploads/MI1141-%C4%90a%CC%A3i-so%CC%82%CC%81-Nho%CC%81m-1.zip. |
| [5] | X. Zhai, "Localization and Change Point Detection using GPS Data". |
| [6] | Trần Bách, Lưới điện và hệ thống điện, Nhà xuất bản Khoa học Kỹ thuật, 2004. |
| [7] | Abe Masayuki, “A Practical Approach to Accurate Fault Location on Extra High Voltage Teed Feeders,” *IEEE Transaction on Power Delivery,* pp. 159-168, 1995. |
| [8] | Microsoft, "Add citations in a Word document," 2017. |
| [9] | T. Đ. Quỳ, "Chương 2 Biến ngẫu nhiên và luật phân phối xác suất 3.1.Kỳ vọng," in *Xác suất thống kê*. |
| [10] | T. Đ. Quỳ, "Chương 2 Biến ngẫu nhiên và luật phân phối xác suất 3.2.Phương sai," in *Xác suất thống kê*. |

# PHỤ LỤC

|  |  |  |
| --- | --- | --- |
| ***Kí hiệu***  Bảng 2.1 | ***Ý nghĩa*** | ***Kí hiệu thay thế*** |
|  | Vecto trạng thái được ước tính tại thời điểm n của bước dự đoán | x |
|  | Vecto trạng thái được ước tính tại thời điểm n-1 của bước cập nhật | x\_pred |
|  | Vecto đầu vào điều khiển dùng để ước tính trạng thái tại bước dự đoán. | v\_wls |
|  | Ma trận chuyển trạng thái | F |
|  | Ma trận điều khiển | G |
|  | Ma trận hiệp phương sai tại thời điểm n-1 của bước cập nhật | P |
|  | Ma trận hiệp phương sai tại thời điểm n của bước dự đoán | P\_pred |
|  | Ma trận hiệp phương sai nhiễu quá trình | Q |
|  | Vecto trạng thái ước tính(đầu ra hệ thống) | x |
|  | Vecto đo lường tại thời điểm n | z |
|  | Vecto phần dư đo lường | y |
|  | Ma trận đo lường | H |
|  | Ma trận đơn vị | I |
|  | Độ lợi Kalman (Kalman Gain) | K |
|  | Ma trận hiệp phương sai nhiễu đo lường ứng với | R |

**Giải thích code libKF.py**

**Khởi tạo lớp KalmanFilter**

A screenshot of a computer code

Description automatically generated

**\_\_init\_\_ method**: Phương thức khởi tạo của **lớp KalmanFilter**, được gọi khi một đối tượng của lớp này được tạo ra.

* **self.sigma\_mahalanobis**: Ngưỡng Mahalanobis để phát hiện giá trị ngoại lai, được đặt là 30.0.
* **self.F**: Ma trận chuyển đổi trạng thái (transition matrix), được khởi tạo là ma trận đơn vị kích thước 3x3.
* **self.G**: Ma trận điều khiển (control matrix), cũng được khởi tạo là ma trận đơn vị kích thước 3x3.
* **self.H**: Ma trận đo lường (measurement matrix), được khởi tạo là ma trận đơn vị kích thước 3x3.
* **self.I**: Ma trận đơn vị (identity matrix) kích thước 3x3, dùng trong các phép tính ma trận.

**Hàm predict (Bước dự đoán)**

A screenshot of a computer code

Description automatically generated

**Mô tả**: Thực hiện bước dự đoán trong bộ lọc Kalman.

**Tham số**:

* **x**: Trạng thái ban đầu hoặc trạng thái ước lượng.
* **u**: Vận tốc hoặc điều khiển đầu vào.
* **P**: Ma trận hiệp phương sai của trạng thái.
* **Q**: Ma trận hiệp phương sai của nhiễu quá trình.

**Trả về**:

* **x\_pred**: Trạng thái dự đoán.
* **P\_pred**: Ma trận hiệp phương sai dự đoán.

**Thực hiện**:

* **x\_pred**: Trạng thái dự đoán được tính bằng cách áp dụng ma trận chuyển đổi trạng thái F và ma trận điều khiển G lên trạng thái x và vận tốc u.
* **P\_pred**: Ma trận hiệp phương sai dự đoán được tính bằng cách nhân ma trận chuyển đổi trạng thái F với ma trận hiệp phương sai trạng thái P, sau đó cộng ma trận hiệp phương sai của nhiễu quá trình Q.

**Hàm update (Bước cập nhật)**

**A computer screen shot of a computer screen

Description automatically generated with medium confidence**

**Mô tả**: Thực hiện bước cập nhật trạng thái trong bộ lọc Kalman.

**Tham số**:

* **x\_pred**: Trạng thái dự đoán từ bước dự đoán.
* **P\_pred**: Ma trận hiệp phương sai dự đoán từ bước dự đoán.
* **z**: Đo lường quan sát.
* **R**: Ma trận hiệp phương sai của nhiễu đo lường.

**Trả về**:

* **x**: Trạng thái ước lượng sau khi cập nhật.
* **P**: Ma trận hiệp phương sai sau khi cập nhật.

**Thực hiện**:

* **y**: Sự khác biệt giữa giá trị đo lường z và trạng thái dự đoán x\_pred.
* **K**: Hệ số Kalman (Kalman Gain), xác định mức độ tin tưởng vào giá trị đo lường.
* **x**: Trạng thái ước lượng sau khi cập nhật, được tính bằng cách áp dụng hệ số Kalman K lên sự khác biệt y.
* **P**: Ma trận hiệp phương sai sau khi cập nhật, được tính bằng cách điều chỉnh ma trận hiệp phương sai dự đoán P\_pred.

**Hàm kalman\_filter** (Kalman Filter)

A screenshot of a computer program

Description automatically generated

**Mô tả**: Thực hiện toàn bộ quá trình lọc Kalman, bao gồm cả dự đoán và cập nhật trạng thái.

**Tham số**:

* **x\_wls**: Trạng thái ban đầu hoặc trạng thái ước lượng từ phương pháp bình phương tối thiểu.
* **v\_wls**: Vận tốc từ phương pháp bình phương tối thiểu.
* **cov\_zs**: Ma trận hiệp phương sai của đo lường.
* **cov\_vs**: Ma trận hiệp phương sai của vận tốc.

**Trả về**:

* **x\_kf**: Các trạng thái ước lượng sau khi áp dụng lọc Kalman.
* **P\_kf**: Các ma trận hiệp phương sai tương ứng với các trạng thái ước lượng.

**Thực hiện**:

* **vs**: Tính vận tốc trung bình giữa các bước thời gian.
* **n, dim\_x**: Số lượng điểm dữ liệu và kích thước của trạng thái.
* **x**: Trạng thái ban đầu.
* **P**: Ma trận hiệp phương sai ban đầu.
* **x\_kf** và **P\_kf**: Các mảng để lưu trữ trạng thái và ma trận hiệp phương sai cho mỗi bước thời gian.
* **Vòng lặp for**: Lặp qua từng cặp (u, z) trong vs và x\_wls.
* **Bước dự đoán**: Gọi phương thức predict để tính trạng thái và ma trận hiệp phương sai dự đoán.
* **Kiểm tra ngoại lai**: Tính khoảng cách Mahalanobis d và so sánh với ngưỡng sigma\_mahalanobis.
* **Bước cập nhật**: Nếu d < sigma\_mahalanobis, gọi phương thức update để cập nhật trạng thái và ma trận hiệp phương sai. Nếu không, điều chỉnh ma trận hiệp phương sai P.
* **Lưu trữ kết quả**: Lưu trữ trạng thái và ma trận hiệp phương sai cho mỗi bước thời gian.

**Giải thích code**

****

* **zs.shape** là một đối tượng zs truy cập vào thuộc tính shape là một mảng 2 chiều.
* **n** là số lượng vecto.
* **dim\_x** là số chiều của mỗi vecto x.

**Giải thích code:**

**Việc tính giá trị trung bình của vận tốc giúp cho dữ liệu được làm mịn hơn !**

A computer code with text

Description automatically generated with medium confidence

**v\_wls[:-1]:**

* v\_wls là một mảng NumPy.
* v\_wls[:-1] lấy tất cả các phần tử v\_wlst rừ phần tử cuối cùng.

**v\_wls[1:]:**

* v\_wls[1:] lấy tất cả các phần tử v\_wls trừ phần tử đầu tiên.

**(v\_wls[:-1] + v\_wls[1:])/2:**

* Biểu thức này tính trung bình của từng cặp phần tử tiếp giáp trong v\_wls.
* Kết quả sẽ là một mảng mới mà mỗi phần tử là trung bình của hai phần tử liền kề từ v\_wls.

**np.zeros([1, 3]):**

* Tạo một mảng NumPy với kích thước (1, 3) và tất cả các phần tử bằng 0.

**np.vstack([...]):**

* np.vstack()là một hàm trong NumPy để xếp chồng các mảng theo chiều dọc (tức là thêm các hàng mới).
* Trong trường hợp này, nó sẽ chồng mảng [[0, 0, 0]] lên trên kết quả mảng của (v\_wls[:-1] + v\_wls[1:])/2.

**Ví dụ minh họa:**

Giả sử v\_wls là một mảng 2 chiều:

A screenshot of a computer code

Description automatically generated

Khi áp dụng đoạn code sau:



Kết quả sẽ là:

* v\_wls[:-1] có giá trị là:

A number with different numbers

Description automatically generated with medium confidence

* v\_wls[1:] có giá trị là:

A number with blue and black text

Description automatically generated with medium confidence

* (v\_wls[:-1] + v\_wls[1:])/2 có giá trị là:

A number of numbers on a white background

Description automatically generated

* np.zeros([1, 3]) có giá trị là:



* np.vstack([np.zeros([1, 3]), (v\_wls[:-1] + v\_wls[1:])/2]) có giá trị là:

A number with numbers in the middle

Description automatically generated with medium confidence

Như vậy, mảng **v** sẽ bao gồm một hàng số không ở đầu, tiếp theo là trung bình của các cặp tử liền kề từ **v\_wls**.

**Giải thích code:**

****

* **enumerate()**:
* Hàm enumerate() được sử dụng để thêm một chỉ số (index) vào mỗi phần tử của iterable (trong trường hợp này là zip(vs, x\_wls)).
* Kết quả của enumerate() là một cặp (index, value) cho mỗi phần tử trong iterable.
* **zip(vs, x\_wls)**:
* Hàm zip() kết hợp các phần tử từ hai (hoặc nhiều) iterables (trong trường hợp này là vs và x\_wls) thành các cặp.
* Kết quả của zip(vs, x\_wls) là một iterable chứa các cặp (u, z), trong đó u là một phần tử từ vs và z là một phần tử tương ứng từ x\_wls.
* **for i, (u, z) in enumerate(zip(vs, x\_wls)):**:
* Vòng lặp for sẽ lặp qua mỗi cặp (u, z) trong iterable được tạo bởi zip(vs, x\_wls).
* Biến i sẽ là chỉ số của cặp (u, z) hiện tại, u là phần tử từ vs, và z là phần tử tương ứng từ x\_wls.