|  |  |  |
| --- | --- | --- |
|  | TRƯỜNG ĐẠI HỌC BÁCH KHOA  KHOA CÔNG NGHỆ THÔNG TIN |  |

BÁO CÁO

PBL5 - ĐỒ ÁN KỸ THUẬT MÁY TÍNH

HỆ THỐNG NHẬN DIỆN VẬT CẢN VÀ PHÁT HIỆN NGƯỜI THÂN CHO NGƯỜI KHIẾM THỊ

Cán bộ doanh nghiệp hướng dẫn: Nguyễn Ngọc Nam

Giảng viên đồng hướng dẫn: ThS. Bùi Thị Thanh Thanh

|  |  |
| --- | --- |
| STT NHÓM: 26  HỌ VÀ TÊN SINH VIÊN | LỚP HỌC PHẦN ĐỒ ÁN |
| Nguyễn Nhật Tùng | 18.15A |
| Trần Anh Duy | 18.15B |
| Lê Hữu Long | 18.15A |
| Nguyễn Thành Vinh | 18.15A |

ĐÀ NẴNG, 07/2021

TÓM TẮT ĐỒ ÁN

Nhằm hỗ trợ người khiếm thị tốt hơn trong việc đi lại trên đường phố đô thị, tăng tính chủ động của người khiếm thị trong việc đi lại, và giúp người thân giám sát được vị trí của thiết bị qua Internet, nhóm tiến hành nghiên cứu giải pháp thông minh có ứng dụng học máy, thiết bị nhúng và IoT. Mô hình học máy được train dựa theo mô hình YOLOv3 tiny có precision và recall lần lượt là 0.7 và 0.37; khi được chạy trên thiết bị Raspberry Pi model 4B thời gian suy diễn trung bình là 960ms, chu kỳ xử lý trung bình của hệ thống là tối thiểu 1532ms. Kết quả đạt được cho thấy hệ thống đã có thể xử lý trong thời gian thực, với kết quả nhận diện chưa đủ nhiều nhưng có độ chính xác tương đối cao.

BẢNG PHÂN CÔNG NHIỆM VỤ

|  |  |  |
| --- | --- | --- |
| Sinh viên thực hiện | Các nhiệm vụ | Tự đánh giá theo 3 mức  (Đã hoàn thành/Chưa hoàn thành/Không triển khai) |
| Nguyễn Nhật Tùng  Lê Hữu Long | 1. Nghiên cứu kiến thức nền tảng về học máy. 2. Thu thập, tạo bộ dữ liệu train, test. 3. Tạo và train các mô hình học máy. 4. Lập trình các hàm API nhận diện trên ảnh để sử dụng trên thiết bị nhúng. | 1. Đã hoàn thành. 2. Đã hoàn thành. 3. Đã hoàn thành. 4. Đã hoàn thành. |
| Trần Anh Duy | 1. Xây dựng web server để hiện thị tọa độ thiết bị vi xử lý. 2. Nhận thông tin vị trí địa lí của thiết bị vi xử lý. 3. Hiển thị vị trí của thiết bị trên bản đồ. | 1. Đã hoàn thành. 2. Đã hoàn thành. 3. Đã hoàn thành. |
| Nguyễn Thành Vinh | 1. Thiết lập Raspberry Pi. 2. Lập trình module chụp và lưu ảnh. 3. Lập trình module nhận diện trên ảnh chụp được. 4. Lập trình module phát tín hiệu thông báo ra loa. 5. Lập trình module lấy tọa độ GPS và gửi tọa độ tới server. | 1. Đã hoàn thành. 2. Đã hoàn thành. 3. Đã hoàn thành. 4. Đã hoàn thành. 5. Đã hoàn thành. |

MỤC LỤC

[1. Giới thiệu 5](#_Toc79870943)

[2. Giải pháp 6](#_Toc79870944)

[2.1. Giải pháp về phần cứng và truyền thông 6](#_Toc79870945)

[2.1.1. Chức năng chụp ảnh từ camera và thông báo ra loa 6](#_Toc79870946)

[2.1.2. Chức năng gửi tọa độ GPS từ thiết bị tới máy chủ 8](#_Toc79870947)

[2.2. Giải pháp về phần mềm 9](#_Toc79870948)

[2.2.1. Chức năng nhận diện vật thể 9](#_Toc79870949)

[2.2.2. Chức năng hiển thị tọa độ của thiết bị 10](#_Toc79870950)

[3. Kết quả 11](#_Toc79870951)

[4. Kết luận 14](#_Toc79870952)

[5. Danh mục tài liệu tham khảo 14](#_Toc79870953)

# Giới thiệu

Năm 2020, trên thế giới có khoảng 49.1 triệu người bị mù, tăng 42.8% so với năm 1990 [1]. Còn ở Việt Nam hiện nay có khoảng 2 triệu người bị mù và thị lực kém [2], sau đây gọi tắt là người khiếm thị.

Để khắc phục khó khăn đi lại của người khiếm thị, nhiều giải pháp đã được đưa ra. Các giải pháp cổ điển như dùng gậy dò đường dành cho người khiếm thị hay cần người hỗ trợ. Cách dùng gậy dò đường thì không thể phát hiện trước các vật cản ở xa dẫn đến có thể không kịp tránh né các vật cản di chuyển tốc độ cao, ngoài ra cũng không thể phát hiện chính xác được nhiều loại vật thể; cách dùng người thân hỗ trợ thì họ không thể luôn có mặt giúp đỡ. Với sự phát triển mạnh của trí tuệ nhân tạo, hay học máy nói riêng trong thời gian gần đây, trên thế giới đã có nhiều công trình nghiên cứu ứng dụng chúng để chế tạo thiết bị thông minh hỗ trợ việc đi lại cho người khiếm thị với độ chính xác cao và nhiều tính năng hữu ích: nhận diện vật cản [3], cảnh báo khoảng cách gần, đọc văn bản [4], thông báo vật thể bằng âm thanh 3D [5],… Trong khi đó các công trình nghiên cứu khoa học tương tự vẫn chưa được thực hiện nhiều ở trong nước. Các giải pháp đã nghiên cứu thì chưa phát hiện được nhiều vật thể [6] hoặc mức độ hỗ trợ còn hạn chế [7].

Nhằm hỗ trợ người khiếm thị tốt hơn trong việc đi lại trên đường phố đô thị, tăng tính chủ động của người khiếm thị trong việc đi lại, và giúp người thân giám sát được vị trí của thiết bị qua Internet, nhóm tiến hành nghiên cứu giải pháp thông minh có ứng dụng học máy, thiết bị nhúng và IoT giúp nhận diện vật cản và người thân (sau đây gọi tắt là vật cản) qua camera và thông báo vật cản nhận diện được qua loa, đồng thời liên tục gửi vị trí GPS của thiết bị tới máy chủ.

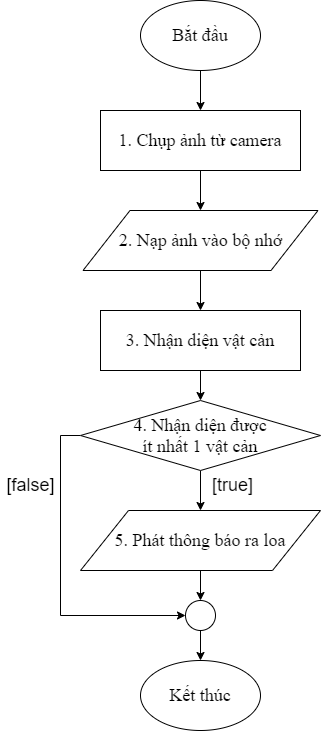
# Giải pháp

Giải pháp về phần cứng và truyền thông

* + 1. Chức năng chụp ảnh từ camera và thông báo ra loa
* Các linh kiện sử dụng:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Thứ tự | Tên linh kiện | Thông số đầu vào | Thông số đầu ra | Nguyên tắc hoạt động |
| 1 | Loa áp điện (buzzer) | * 3.3V hoặc 5V điện DC. * Cường độ dòng điện tối đa: 30MA/5VDC. | * Tần số cộng hưởng: 2500Hz (± 300 Hz). | Ứng dụng hiệu ứng áp điện: Thay đổi tần số điện áp đưa vào loa làm cho viên thạch anh bên trong loa thay đổi hình dạng rất nhanh và phát ra âm thanh. |
| 2 | Pi camera module v1.3 | * Dòng điện 1 chiều 5V – 2A. | * Chế độ ảnh tĩnh 5MP. * Độ phân giải của cảm biến: 2592 × 1944 pixels. * Định dạng ảnh: JPEG, RAW, GIF, BMP, PNG, YUV420, RGB888. | Khi module được kích hoạt, các cảm biến của camera tạo ra dòng điện có cường độ phụ thuộc vào lượng ánh sáng chúng nhận được. Cường độ dòng điện của từng cảm biến được chuyển thành và lưu dưới 1 định dạng ảnh hoặc raw. |

* Sơ đồ khối:



Hình 1. Sơ đồ khối chức năng chụp ảnh từ camera và thông báo vật thể nhận diện được ra loa.

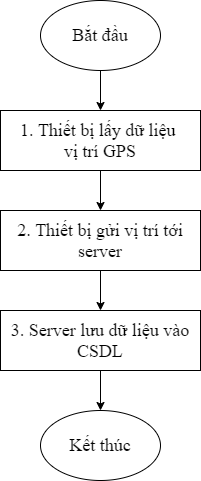
* Mô tả nguyên lý hoạt động:

1. Gọi hàm chụp ảnh từ camera và lưu tại bộ nhớ trong (thẻ microSD).
2. Nạp ảnh vừa chụp vào bộ nhớ RAM của thiết bị vi xử lý.
3. Truyền dữ liệu ảnh vào module thực hiện chức năng nhận diện vật cản và lấy về dữ liệu dự đoán.
4. Kiểm tra trong dữ liệu trả về nếu phát hiện ít nhất 1 vật cản thì thực hiện bước 5, ngược lại thì kết thúc.
5. Cho mỗi dữ liệu vật cản nhận diện được, phát thông báo đặc trưng cho lớp vật cản đó ra loa.
   * 1. Chức năng gửi tọa độ GPS từ thiết bị tới máy chủ

* Các linh kiện sử dụng:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Thứ tự | Tên linh kiện | Thông số đầu vào | Thông số đầu ra | Nguyên tắc hoạt động |
| 1 | Module GPS Neo-6M | * Dòng điện 1 chiều 2.7V ~ 3.6V – 45mA. | * Serial Baud Rate 4800-230400 (default 9600). * Thông tin ngày giờ, vị trí (kinh độ, vĩ độ), … được gửi theo giao thức NMEA. | Module đóng vai trò là receiver. Khi được kích hoạt thì nó liên lạc với ít nhất 3 vệ tinh để tính toán ra vị trí địa lý của module, bao gồm kinh độ, vĩ độ, độ cao, vận tốc, ngày giờ và một số thông tin khác được định dạng theo giao thức NMEA. Dữ liệu nhận được sau đó được truyền vào thiết bị vi xử lý bằng cổng serial. |

* Sơ đồ khối:



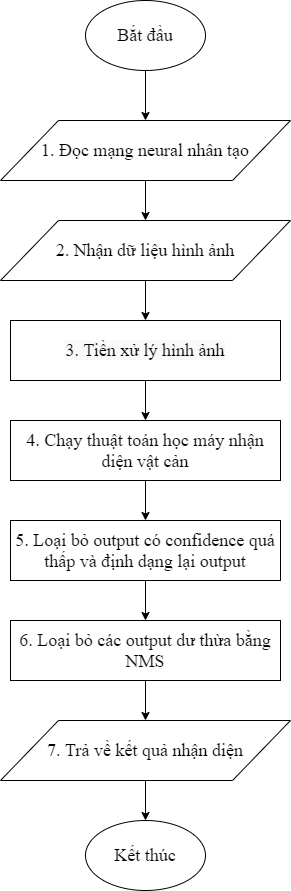
Hình 2. Sơ đồ khối chức năng gửi tọa độ GPS từ thiết bị tới máy chủ.

* Mô tả nguyên lý hoạt động:

1. Thiết bị nhúng thu dữ liệu vị trí GPS từ module GPS.
2. Thiết bị gửi dữ liệu đó tới server bằng giao thức HTTP, sử dụng chức năng kết nối wifi tích hợp sẵn trên thiết bị Raspberry Pi.
3. Server nhận được dữ liệu từ thiết bị và lưu vào CSDL.

Giải pháp về phần mềm

* + 1. Chức năng nhận diện vật thể
* Sơ đồ khối:

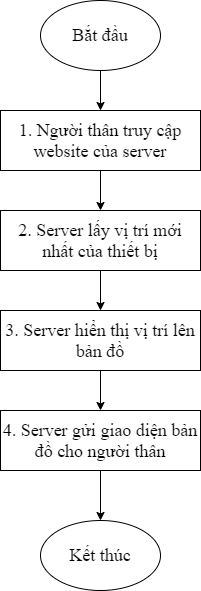


Hình 3. Sơ đồ khối chức năng nhận diện vật thể.

* Mô tả nguyên lý hoạt động:

1. Chương trình đọc thông tin về tổ chức và trọng số của mạng nơ-ron nhân tạo dựa vào mô hình học máy đã được huấn luyện sẵn.
2. Nhận vào tham số mảng dữ liệu hình ảnh cần nhận diện vật cản.
3. Dữ liệu hình ảnh được resize thành mảng 3 chiều kích thước 320x320x3.
4. Chạy thuật toán học máy với dữ liệu hình ảnh đã qua bước tiền xử lý bằng mạng nơ-ron đã huấn luyện và lấy dữ liệu trả về ở output layer của mạng.
5. Lặp qua từng detection, tìm confidence lớn nhất và loại bỏ detection này nếu confidence nhỏ hơn ngưỡng cho phép, ngược lại tiến hành định dạng lại detection này để thực hiện Non-Max Supression (NMS) ở bước 6.
6. Thực hiện thuật toán NMS để loại bỏ các detection dư thừa (nếu chúng có cùng class và có giá trị IoU (Intersection over Union) cao hơn ngưỡng cho phép).
7. Trả về kết quả nhận diện gồm danh sách các detection, danh sách các confidence và danh sách các class ID.
   * 1. Chức năng hiển thị tọa độ của thiết bị

* Sơ đồ khối:



Hình 4. Sơ đồ khối chức năng hiển thị tọa độ GPS của thiết bị từ website của máy chủ.

* Mô tả nguyên lý hoạt động:

1. Người thân của người sử dụng thiết bị truy cập vào địa chỉ website của server bằng trình duyệt.
2. Khi nhận được request, server lấy vị trí mới nhất của thiết bị trong cơ sở dữ liệu.
3. Server hiển thị vị trí đó lên bản đồ bằng Google Maps API.
4. Server gửi giao diện bản đồ dạng file .html cho người thân.

# Kết quả

* Web server được lập trình theo kiến trúc ứng dụng web động (dynamic web application), theo đó giao diện hiển thị cho người dùng sẽ được server tạo lại cho mỗi request. Để triển khai tính năng cập nhật tọa độ người thân liên tục cho người dùng, nhóm sử dụng công nghệ web socket để server và client có thể giữ kết nối và gửi dữ liệu cho nhau. Nhóm sử dụng “ExpressJS” web framework [8] và thư viện “EJS” [9] để xây dựng ứng dụng web động; thư viện “ws” [10] được sử dụng để thiết lập web socket giữa server và client.
* Dữ liệu huấn luyện mô hình học máy bao gồm 4 lớp: “person”, “car”, “motorcycle” và “vinh”, với “vinh” là lớp người thân. Số lượng ảnh là 7680 tấm, trong đó có mỗi lớp “person”, “car”, “motorcycle” chiếm 2000 tấm và được lấy từ bộ dữ liệu COCO [11], số ảnh 1608 tấm còn lại thuộc về lớp “vinh” được thu thập bằng cách tách các khung ảnh đặc trưng từ bản ghi video màu, có độ phân giải 1280x720. Bộ dữ liệu được tách theo tỷ lệ 8:2 để chia ra bộ dữ liệu huấn luyện (train) và bô dữ liệu kiểm tra (test).

|  |  |  |  |
| --- | --- | --- | --- |
| ***Tỷ lệ bộ dữ liệu huấn luyện/kiểm tra = 8/2*** | | | |
| **Tên lớp** | **Số ảnh huấn luyện** | **Số ảnh kiểm tra** | **Tổng số ảnh** |
| Person | 1600 | 400 | 2000 |
| Car | 1600 | 400 | 2000 |
| Motorcycle | 1600 | 400 | 2000 |
| Vinh | 1344 | 322 | 1608 |
| **Tổng** | **6086** | **1522** | **7608** |

Bảng 1. Cấu trúc bộ dữ liệu sử dụng.

* Mô hình học máy được huấn luyện theo mô hình YOLOv3 tiny (một phiên bản được thu nhỏ từ mô hình gốc YOLOv3 [12]), được train bằng cách sử dụng Darknet framework [13], train trên Google Colab và backup của mô hình train được lưu vào Google Drive.
* Mô hình học máy sau khi train được đánh giá bằng bộ dữ liệu kiểm thử (test) và đạt kết quả là độ chính xác (precision) đạt 0.7, recall đạt 0.37 ở ngưỡng IoU=0.3. Sử dụng độ đo mean average precision (mAP) cho kết quả là 0.633643 với ngưỡng IoU=0.3.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Ngưỡng IoU** | **Person** | **Car** | **Motorcycle** | **Vinh** |
| 0.3 | 0.5131 | 0.4752 | 0.5544 | 0.9919 |
| 0.5 | 0.3617 | 0.3378 | 0.3977 | 0.9673 |

Bảng 2. Độ chính xác trung bình (average precision) cho từng lớp ở 2 ngưỡng IoU.

|  |  |  |  |
| --- | --- | --- | --- |
| Ngưỡng IoU | Precision | Recall | mAP |
| 0.3 | 0.7 | 0.37 | 0.633643 |
| 0.5 | 0.64 | 0.33 | 0.516127 |

Bảng 3. Các đơn vị đo đánh giá toàn bộ mô hình.

* Thời gian suy diễn trung bình (average inference time) đạt 23.291ms, xấp xỉ 43 FPS trên laptop và đạt 960ms, xấp xỉ 1.042 FPS trên máy tính nhúng Raspberry Pi model 4B (sau đây gọi tắt là Pi). Chu kỳ trung bình tối thiểu một vòng lặp xử lý của hệ thống trên thiết bị Pi là 1532ms, gồm thời gian chụp ảnh và thời gian suy diễn; với mỗi vật cản phát hiện được, chu kỳ xử lý của hệ thống tăng thêm 500ms để gửi thông báo âm thanh ra loa.

|  |  |  |
| --- | --- | --- |
| **Thời gian suy diễn (ms)** | **FPS (khung xử lý/s)** | **Chu kỳ xử lý tối thiểu (ms)** |
| 960 | 1.042 | 1532 |

Bảng 4. Các đơn vị đo đánh giá tốc độ xử lý của chức năng nhận diện vật cản và của hệ thống.

* Để thông báo dữ liệu nhận diện được ra loa áp điện, nhóm ánh xạ mỗi lớp vật cản thành một tần số âm, mỗi âm thanh được phát ra liên tiếp nhau trong thời gian 0.5 giây/âm thanh. Cách ánh xạ được cho theo bảng dưới đây.

|  |  |
| --- | --- |
| *Tần số tối thiểu: 110 (nốt La quãng 2 - A2)*  *Tần số tối đa: 1975.5 (nốt Si quãng 6 - B6)* | |
| **Tên lớp vật cản** | **Tần số âm phát ra loa** |
| Person | 110 |
| Car | 731 |
| Motorcycle | 1353 |
| Vinh | 1975 |

* Hệ thống được kiểm thử bằng cách: người dùng A đeo thiết bị vi xử lý và đi bộ trên đoạn đường 100m, đồng thời người dùng B truy cập trang web của hệ thống trong mạng WLAN để theo dõi sự thay đổi tọa độ của người A trên bản đồ của trang web. Kết quả đạt được là thiết bị trên người A có thể nhận diện được các vật cản và phát thông báo chính xác ra loa, tọa độ GPS của người A cũng được gửi tới web server và người B theo dõi được di chuyển của người A trên suốt quãng đường 100m với sai số vị trí theo chiều ngang xấp xỉ 2.5 m.

# Kết luận

Mô hình học máy được train dựa theo mô hình YOLOv3 tiny có precision và recall lần lượt là 0.7 và 0.37; khi được chạy trên thiết bị Raspberry Pi model 4B thời gian suy diễn trung bình là 960ms, chu kỳ xử lý trung bình của hệ thống là tối thiểu 1532ms. Kết quả đạt được cho thấy hệ thống đã có thể xử lý trong thời gian thực, với kết quả nhận diện chưa đủ nhiều nhưng có độ chính xác tương đối cao.

Kết quả kiểm thử cho thấy hệ thống có tính IoT tốt, chức năng cơ bản của từng module được đảm bảo. Tuy nhiên hệ thống có nhiều khả năng phát triển trong tương lai nếu có thêm kinh phí và thời gian phát triển: phát hiện tường ở cự ly gần bằng sóng siêu âm, cải thiện mô hình nhận diện bằng bộ dữ liệu mới được chọn lọc chất lượng và đầy đủ hơn.

# Danh mục tài liệu tham khảo

1. Rupert R A Bourne, Jaimie Adelson, Seth Flaxman, Paul Briant, Michele Bottone, Theo Vos, Kovin Naidoo, Tasanee Braithwaite, Maria Cicinelli, Jost Jonas, Hans Limburg, Serge Resnikoff, Alex Silvester, Vinay Nangia, Hugh R Taylor; *Global Prevalence of Blindness and Distance and Near Vision Impairment in 2020: progress towards the Vision 2020 targets and what the future holds*.. Invest. Ophthalmol. Vis. Sci. 2020;61(7):2317.
2. Mai, K., (2019). *Việt Nam có khoảng 2 triệu người mù và thị lực kém*. Sức khỏe và Đời sống. https://suckhoedoisong.vn/khoang-2-trieu-nguoi-mu-va-thi-luc-kem-1-3-nguoi-ngheo-khong-co-tien-dieu-tri-tim-lai-anh-sang-n164407.html.
3. Joshi, R. C., Yadav, S., Dutta, M. K., & Travieso-Gonzalez, C. M. (2020). *Efficient Multi-Object Detection and Smart Navigation Using Artificial Intelligence for Visually Impaired People*. Entropy (Basel, Switzerland), 22(9), 941.
4. Jiang, R., Lin, Q., & Qu, S. (2016). *Let blind people see: real-time visual recognition with results converted to 3D audio*. Report No. 218, Standord University, Stanford, USA.
5. Mandhala, V. N., Bhattacharyya, D., Vamsi, B., & Thirupathi Rao, N. (2020). *Object detection using machine learning for visually impaired people*. International Journal of Current Research and Review, 12(20), 157-167.
6. Quoc Hung, N., Thi Thanh Hai, T., Hai, V., Van Nam, H. & Quang Hoan, N. (2016). *Phát hiện và ước lượng khoảng cách vật cản, ứng dụng trọ giúp dẫn đường cho người khiếm thị*. Tạp chí khoa học công nghệ thông tin và truyền thông, Hà Nội, Việt Nam.
7. Van Chuong, N. (2019). Nghiên cứu chế tạo thiết bị cảnh báo trước chướng ngại vật trong hoạt động đi lại của người khiếm thị. Viện công nghệ Thông tin và Truyền thông CDIT, Hà Nội, Việt Nam.
8. Kononenko, K. (2017, November 3). *Going out to eat and understanding the basics of Express.js.* <https://www.freecodecamp.org/news/going-out-to-eat-and-understanding-the-basics-of-express-js-f034a029fb66/>.
9. Sev, C. (2021, May 5). *How To Use EJS to Template Your Node Application*. <https://www.digitalocean.com/community/tutorials/how-to-use-ejs-to-template-your-node-application>.
10. Franchetti, J. (2021, July 22). *WebSockets and Node.js - testing WS and SockJS by building a web app*. <https://ably.com/blog/web-app-websockets-nodejs>.
11. Lin T.-Y., Maire M., Belongie S., Hays J., Perona P., Ramanan D., Dollár P., Zitnick C.L. Bioinformatics Research and Applications. Volume 8693. Springer Science and Business Media LLC; Berlin, Germany: 2014. *Microsoft COCO: Common Objects in Context*; pp. 740–755.
12. Redmon, J., & Farhadi, A. (2018). *YOLOv3: An Incremental Improvement*. arXiv:1804.02767 [cs].
13. Redmon, J. (2016). *Darknet: Open Source Neural Networks in C*. <http://pjreddie.com/darknet/>.