ĐẠI HỌC QUỐC GIA TP.HCM

TRƯỜNG ĐẠI HỌC BÁCH KHOA

KHOA KHOA HỌC & KỸ THUẬT MÁY TÍNH

LUẬN VĂN TỐT NGHIỆP ĐẠI HỌC

CHƯƠNG TRÌNH HƯỚNG DẪN CHƠI RUBIK

NGÀNH: KHOA HỌC MÁY TÍNH

GVHD: ThS. Trần Giang Sơn

GVPB: ThS. Trần Hồng Tài

---o0o---

SVTH1: Nguyễn Tất Chung - 1810055

SVTH2: Nguyễn Thành Phát - 1810425

TP. HỒ CHÍ MINH, THÁNG 5 NĂM 2022

LỜI CAM ĐOAN

Chúng em xin cam đoan rằng kết quả nghiên cứu được trình bày trong báo cáo luận văn tốt nghiệp này là kết quả học tập và lao động của nhóm dưới sự hướng dẫn của thầy Trần Giang Sơn. Nếu không đúng sự thật, chúng em xin chịu mọi trách nhiệm trước thầy.

Mặc dù đã cố gắng hoàn thành luận văn tốt nghiệp nhưng chắc chắn sẽ không tránh khỏi nhiều thiếu sót. Nhóm em rất mong nhận được sự thông cảm, góp ý và chỉ bảo của quý Thầy.

Tp. Hồ Chí Minh, ngày 15 tháng 5 năm 2022

Sinh viên thực hiện

Nguyễn Tất Chung **Nguyễn Thành Phát**

LỜI CẢM ƠN

Đầu tiên, nhóm em gửi lời cảm ơn đến **Thạc sĩ Trần Giang Sơn** đã cung cấp những ý tưởng về đề tài, chia sẻ tài liệu nghiên cứu và tận tình hướng dẫn trong quá trình học tập và nghiên cứu đề nhóm có thể hoàn thành luận văn.

Em cũng xin chân thành cảm ơn quý thầy cô đang giảng dạy ở khoa Khoa học và Kỹ thuật máy tính trường Đại học Bách Khoa ĐHQG tp HCM. Cảm ơn thầy cô đã luôn tận tâm trong việc truyền đạt kiến thức chuyên môn cũng như những chia sẻ kinh nghiệm quý giá để chúng em trang bị đủ tri thức bước vào đời. Cảm ơn những món quà tri thức vô giá giúp em biết mình còn nhỏ bé đến thế nào để em tiếp tục cố gắng.

Cảm ơn trường Đại học Bách Khoa đã cho em một môi trường học tập và rèn luyện. Bên cạnh vốn kiến thức chuyên môn mà em đã có được, quan trọng hơn hết, đó làem đã học thêm được nhiều bài học quý giá về cách học tập, kỹ năng sống và rèn luyện tinh thần. Cảm ơn những mùa bài tập lớn, mùa thi, những bài thí nghiệm và cả những chương trình, chiến dịch tình nguyện của trường đã nuôi lớn tâm trí em từng ngày. Cảm ơn những kỉ niệm gắng bó với mái trường này, em sẽ không bao giờ có thể trải qua lần nữa. Cảm ơn những người bạn đã cùng tôi đi qua năm tháng ở Bách Khoa, những người bạn sống cùng, học cùng. Những người bạn đã dang đôi tay giúp đỡ khi tôi cần, chong đèn cùng tôi hoàn thành bài tập. Những người bạn đi cùng tôi hay không còn bên cạnh tôi nữa. Cảm ơn các bạn.

Trân trọng.

Tp. Hồ Chí Minh, ngày 15 tháng 5 năm 2022

Sinh viên thực hiện

Nguyễn Tất Chung **Nguyễn Thành Phát**

TÓM TẮT

Rubik là một trò chơi trí tuệ giải khối lập phương thú vị và hấp dẫn nhiều người, đặc biệt là giới trẻ, rất tốt cho trẻ để giải trí đồng thời rèn luyện trí thông minh và tư duy. Tên của trò chơi được đặt theo người phát minh ra nó là Erno Rubik, người Hungary. Ra đời năm 1974, khối Rubik đầu tiên là một khối hình lập phương với 3 x 3 x 3 và 6 màu sắc gồm cam, đỏ, xanh lá, vàng, màu trắng và xanh dương. Nhưng hiện nay khối Rubik rất đa dạng giúp người chơi có nhiều lựa chọn hấp dẫn hơn.

Sự hấp dẫn của khối rubik là thế, nhưng không phải ai cũng dễ dàng tiếp cận và tập luyện thành thạo để giải chúng.

Hiểu được vấn đề đó, áp dụng những kiến thức đồ họa của thư viện OpenGL và thị giác máy tính của thư viện OpenCV, nhóm chúng em đã viết chương trình Hướng dẫn chơi rubik nhằm giúp những người chơi mới dễ dàng tiếp cận và thành thạo bộ môn này.

BỐ CỤC LUẬN VĂN

Luận văn được tổ chức thành năm chương chính bao gồm Giới thiệu, Giải pháp hiện thực, Thiết kế chương trình và một số thuật toán phức tạp, Kết quả hiện thực và đánh giá và cuối cùng là chương Tổng kết.

Chương Giới thiệu sẽ cung cấp cái nhìn toàn cảnh về khối rubik và các thuật toán để giải khối rubik 3x3. Từ đó lựa chọn thuật toán và vạch ra bước đi cụ thể nằm hoàn thành đề tài.

Những thiết kế ban đầu cho chương trình cùng với lí thuyết của một số thuật toán phức tạp được trình bày ở chương Thiết kế chương trình và một số thuật toán phức tạp.

Chương Kết quả hiện thực và đánh giá trình bày kết quả của chương trình và một số những đánh giá chủ quan về chương trình sau khi hoàn thành.

Cuối cùng, chương Tổng kết nêu ra những gì đã làm được, những gì cần phải cải tiến và hướng phát triển của đề tài trong tương lai.

MỤC LỤC

DANH MỤC HÌNH ẢNH

DANH MỤC BẢNG

1. GIỚI THIỆU

1.1. Sự ra đời của khối rubik:

Khối Rubik là đồ chơi hình lập phương với 9 ô vuông nhỏ ở mỗi mặt. Khi mới lấy ra khỏi hộp sản phẩm, tất cả các ô vuông trên từng mặt của khối Rubik đều cùng màu. Mục đích của người chơi là khôi phục lại trạng thái ban đầu này, sau khi họ xoay và xáo trộn khối lập phương theo chiều ngang và chiều dọc nhiều lần. Thoạt nhìn, trò chơi Rubik trông có vẻ đơn giản, nhưng thực tế không phải vậy. Sau vài giờ chơi thử, hầu hết mọi người đều không thể giải khối Rubik nếu không biết công thức từ trước.

Ernö Rubik là người đã chế tạo khối Rubik đầu tiên năm 1974. Nhưng mãi đến năm 1980, loại đồ chơi này mới xuất hiện phổ biến trên thị trường thế giới và trở thành trò chơi trí tuệ được nhiều người yêu thích.

Hình 1-1: Erno Rubik, người sáng chế ra khối Rubik.

Ernö Rubik sinh ra tại thành phố Budapest, Hungary vào ngày 13 tháng 7 năm 1944. Cha của ông là kỹ sư thiết kế tàu lượn. Mẹ ông là nghệ sĩ, kiêm nhà thơ. Rubik đã kết hợp tài năng khác biệt của cha mẹ để trở thành một nhà điêu khắc và kiến trúc sư. Do thích thú với các khái niệm về không gian, Rubik dành thời gian rảnh của mình khi làm giáo sư tại Học viện Nghệ thuật Ứng dụng và Thiết kế ở Budapest để sáng chế ra các đồ dùng, đồ chơi phục vụ việc dạy học, giúp sinh viên có thêm những cái nhìn mới về hình học ba chiều.

Vào mùa xuân năm 1974, Rubik hình dung ra một khối lập phương nhỏ. Mỗi mặt của nó bao gồm các hình vuông có khả năng di chuyển. Đến mùa thu năm 1974, những người bạn đã giúp Rubik tạo ra mô hình gỗ đầu tiên của khối đồ chơi Rubik theo ý tưởng của ông.

Lúc đầu, Rubik chỉ thích quan sát các ô vuông di chuyển khi ông xoay khối Rubik. Tuy nhiên, ông gặp nhiều khó khăn trong lúc cố gắng đưa tất cả các ô vuông cùng màu về một mặt. Không chịu khuất phục, ông đã dành một tháng để xoay khối Rubik theo nhiều cách, cho đến khi sắp xếp lại các màu sắc như ban đầu.

Khi Rubik giới thiệu đồ chơi mới cho những người xung quanh, họ đều vô cùng thích thú và say mê giải câu đố. Rubik nhận ra rằng, món đồ chơi này có thể giúp ông kiếm rất nhiều tiền nếu bán ra thị trường.

Năm 1975, Rubik thực hiện một thỏa thuận với công ty đồ chơi Politechnika ở Hungary để sản xuất hàng loạt các khối Rubik. Năm 1977, khối lập phương nhiều màu sắc lần đầu tiên xuất hiện trong các cửa hàng đồ chơi ở Budapest với tên gọi Büvös Kocka (Khối lập phương Ma thuật). Mặc dù Büvös Kocka là sản phẩm đồ chơi khá thành công ở Hungary, nhưng việc bán nó ra khắp thế giới là một thách thức không nhỏ thời bấy giờ.

Năm 1979, Rubik ký hợp đồng với công ty đồ chơi Ideal Toy nhằm đưa sản phẩm của mình sang Mỹ và một số nước khác ở phương Tây. Ideal Toy đổi tên sản phẩm Büvös Kocka thành "Khối lập phương Rubik" trước khi bày bán rộng rãi nó trong các cửa hàng vào năm 1980. Không lâu sau, khối Rubik trở thành loại đồ chơi phổ biến và được nhiều người ưa chuộng. Nó hấp dẫn cả trẻ em cũng như người lớn.

Những khối Rubik thời kỳ đầu có sáu mặt với các màu khác nhau bao gồm xanh dương, xanh lá cây, cam, đỏ, trắng và vàng. Mỗi mặt có 9 ô vuông xắp xếp theo mô hình lưới 3×3. Trong số 54 ô vuông nhỏ trên khối Rubik, 48 ô vuông có thể di chuyển và 6 ô vuông ở trung tâm mỗi mặt đứng yên.

Tính đến năm 1982, số lượng khối Rubik bán ra thị trường ước tính khoảng 100 triệu. Tuy nhiên, đa số người chơi lúc đó không thể giải khối Rubik nhanh chóng. Nguyên nhân là do số lượng cách sắp xếp trật tự của các mảng màu trên khối Rubik quá lớn, hơn 43 tỷ tỷ cách (chính xác là 43.252.003.274.489.856.000 hoán vị). Một số người

chơi Rubik thiếu kiên nhẫn thậm chí còn đập vỡ khối Rubik để quan sát cấu trúc bên trong. Họ hy vọng sẽ khám phá ra một số bí mật giúp giải câu đố trò chơi.

Để giúp công chúng hiểu nguyên lý hoạt động và công thức giải khối Rubik, hàng chục cuốn sách hướng dẫn đã được xuất bản vào đầu thập niên 1980. Mỗi cuốn sách trình bày những cách đơn giản nhất để người chơi thực hành theo. Cách giải thông dụng nhất do David Singmaster, một nhà toán học người Anh công bố trong cuốn "Notes on Rubik's Magic Cube". Phương pháp của Singmaster là giải khối Rubik từng tầng một, và nó phù hợp với người mới bắt đầu.

Kể từ đó, người chơi bắt đầu thiết lập các kỷ lục về tốc độ và thời gian xoay Rubik.

Năm 1982, Giải vô địch Rubik Quốc tế được tổ chức lần đầu tiên ở Budapest. Đây là cơ hội để những người chơi thi đấu với nhau, xem ai có thể giải khối Rubik nhanh nhất. Năm 2018, tuyển thủ Yusheng Du người Trung Quốc thiết lập kỷ lục thế giới với thành tích giải khối Rubik trong 3,47 giây.

Trong thời kỳ thịnh hành, đồ chơi Rubik xuất hiện ở khắp mọi nơi – từ trường học, trên xe buýt, trong rạp chiếu phim và thậm chí tại nơi làm việc. Hình vẽ khối Rubik cũng được in trên áo phông và áp phích.

Năm 1983, khối Rubik thậm chí còn có chương trình truyền hình riêng gọi là: "Rubik, khối lập phương tuyệt vời". Trong chương trình dành cho trẻ em này, một khối Rubik biết nói với sự giúp đỡ của ba đứa trẻ đã chặn đứng các kế hoạch xấu xa của nhân vật phản diện.

Vấn đề đặt ra là người chơi cần tối thiểu bao nhiêu thao tác xoay để giải bất kỳ khối Rubik nào bị xáo trộn? Năm 2010, Tomas Rokicki và cộng sự phát hiện mọi trạng thái của khối Rubik đều có thể được giải trong 20 bước hoặc ít hơn.

Cho đến nay, người ta đã phát triển khối Rubik thành nhiều phiên bản khác nhau, bao gồm 4 loại chính: 2×2×2 (Khối bỏ túi), 3×3×3 (Khối tiêu chuẩn), 4×4×4 và 5×5×5 (Rubik giáo sư). Gần đây các khối lớn hơn đã xuất hiện trên thị trường như khối 6×6×6 và 7×7×7 (V-Cube 6 và V-Cube 7). Từ khối Rubik lập phương tiêu chuẩn, các nhà sản xuất cũng tạo ra các khối có dạng hình học khác như tứ diện, bát diện, khối 12 mặt, khối 20 mặt, hoặc các khối không lập phương như 2×3×4, 3×3×5, 1×2×3. Với máy tính, họ

thậm chí có thể mô phỏng các khối Rubik lạ trong không gian đa chiều mà người bình thường không thể chế tạo ngoài thực tế.

Hình 1-2: Các rubik có đa dạng các loại

1.2. Lợi ích của việc chơi rubik

Rubik là một bộ môn, một món đồ chơi giải trí đòi hỏi sự tập trung và kiên nhẫn. Có một số lợi ích của việc chơi rubik đến não bộ của chúng ta. Nhưng chơi rubik không phải hề đơn giản vì người chơi phải tập trung để giải được các mặt đúng các màu với nhau.

Cải thiện trí nhớ:

Với những người có "não cá vàng", dễ quên, đây quả là một lợi ích tuyệt vời, vì việc chơi rubik có thể làm cải thiện trí nhớ của họ, vì người chơi cần phải nhớ được rất nhiều công thức, thậm chí trường hợp đặc biệt, để có thể giải thành công khối rubik đó. Người chơi có thể luyện tập thường xuyên để tăng cường cho trí nhớ của mình.

Tập luyện sự kiên nhẫn:

Sự kiên nhẫn của của con người có thể được thử thách khi bạn chơi rubik, đặc biệt là khi không sử dụng đến một công thức nào. Điều này rõ ràng gây khó chịu mà. Nhưng bất kỳ điều gì cũng có thể tập luyện được, sự kiên nhẫn cũng vậy. Thường thì người mới sẽ từ bỏ khối rubik trong vòng 15 phút do họ chỉ mới bắt đầu tập chơi. Do đó, người chơi có thể rèn luyện tính kiên nhẫn hơn trong việc giải quyết vấn đề.

Hình thức giải trí lành mạnh:

Rubik là một trò chơi giải trí rất thích hợp khi con người có thời gian rảnh. Ví dụ, trong lúc đang "xả hơi", "giải lao" trong lúc đi làm hoặc đi học, người chơi có thể thử giải khối rubik vì nó dễ mang theo bên mình. Ngoài ra, khi đang chán nản và không có gì khác để làm, đây là một trò chơi mà người chơi có thể cân nhắc để giết thời gian. Khối rubik có thể đem chúng ta đến bất kỳ nơi đâu trong trí tưởng tượng của mình nên hãy thử nó khi rảnh rỗi.

Duy trì trí nhớ ngắn hạn:

Một lợi ích khác của Rubik là duy trì trí nhớ ngắn hạn. Cũng như trò giải ô chữ và cờ vua, Rubik có những lợi ích tương tự. Đối với những người gặp khó khăn trong việc ghi nhớ điều gì đó không xảy ra thường xuyên hoặc trong ngắn hạn, hãy thử chơi trò này. Điều này rất hiệu quả để giúp cải thiện trí nhớ trong não của chúng ta.

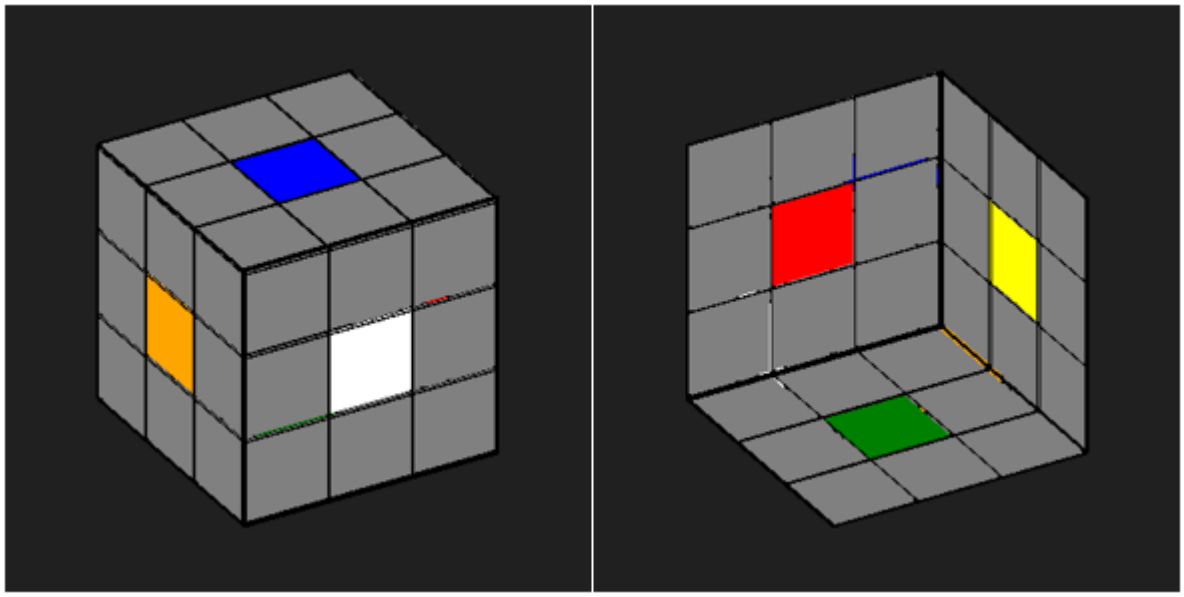
Tăng sự tập trung:

Rubik cũng giúp tăng khả năng tập trung. Hãy để ý cách mà người chơi tập trung lựa chọn cách giải rubik để hoàn thành một cách nhanh nhất. Trò chơi này đòi hỏi sự tập trung cao độ. Nếu chúng ta đang gặp khó khăn trong việc tập trung thì đây vừa là một công cụ giải trí, vừa là giải pháp cho việc tăng khả năng tập trung.

1.3. Tổng quan về khối rubik 3x3 và các ký hiệu

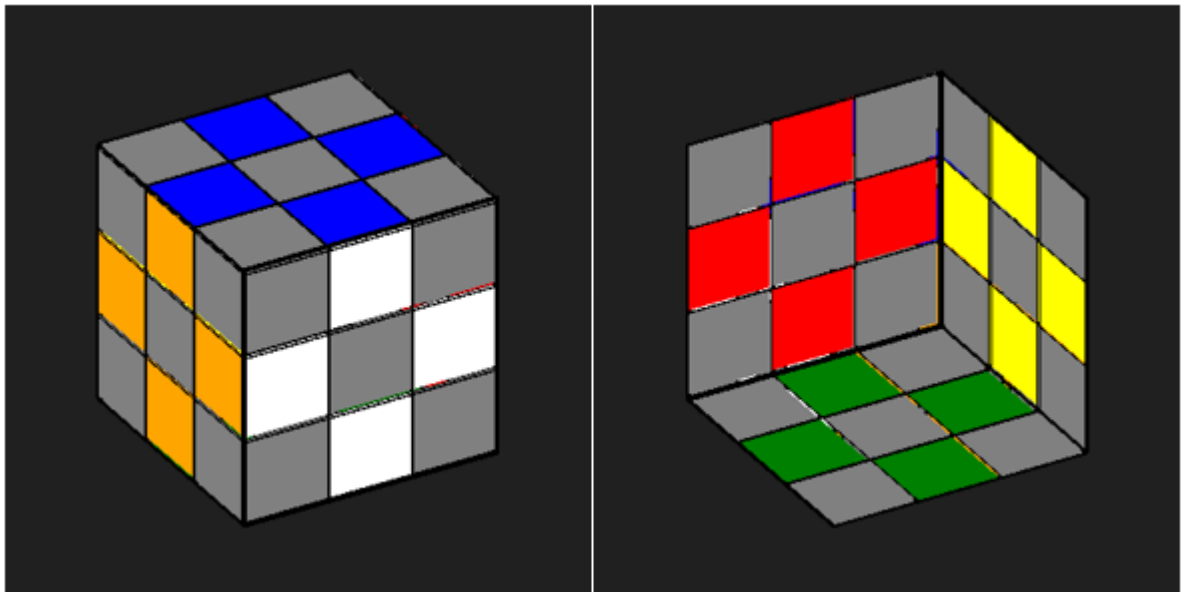
Một khối Rubik có 6 mặt: Trắng, Vàng, Đỏ, Cam, Lục, Lam; gồm 26 viên ghép lại với nhau trong đó có:

6 viên tâm: mỗi viên chỉ có 1 mặt màu, dù bạn quay như thế nào đi nữa thì vị trí của các viên này đều không thay đổi. Như vậy, màu của một viên tâm ở một mặt nào đó cũng chính là màu của cả mặt đó.



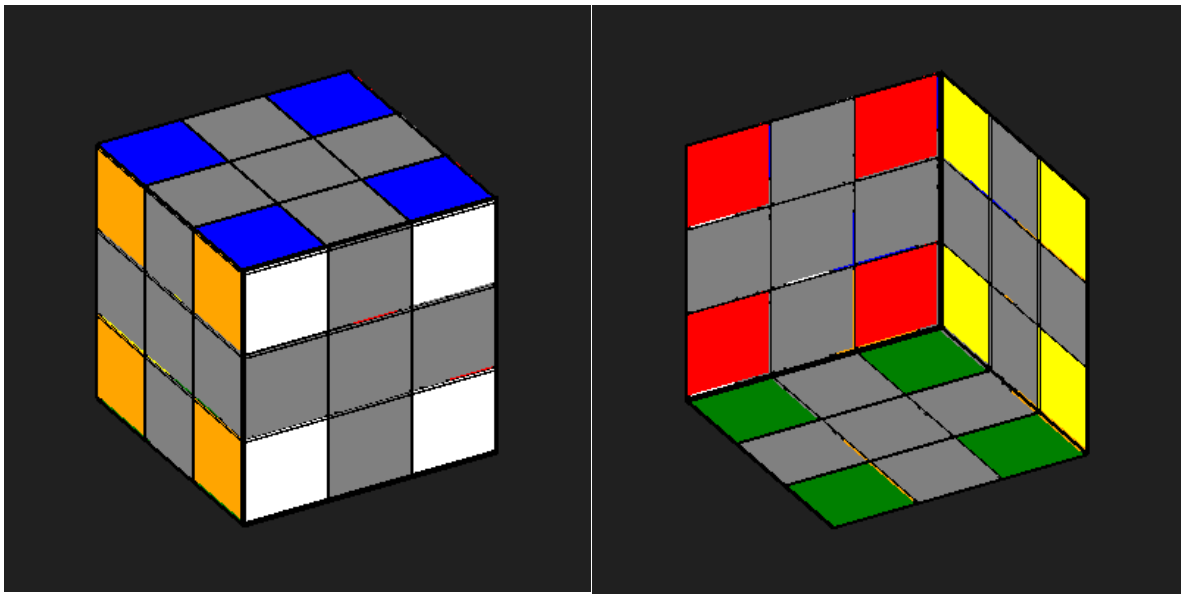
Hình 1-3: Viên tâm của khối rubik

12 viên cạnh: mỗi viên có 2 mặt màu. Các viên này nằm giữa các cạnh của khối Rubik.



Hình 1-4: Viên cạnh của khối rubik

8 viên góc: mỗi viên có 3 mặt màu. Các viên nằm ở các góc của khối Rubik.



Hình 1-5: Viên góc của khối rubik

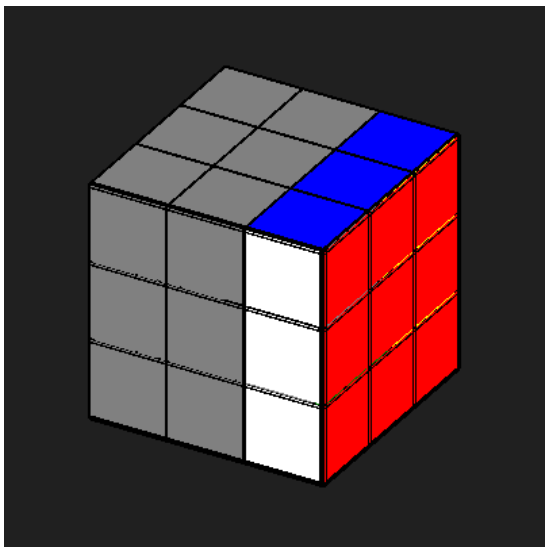
Ta luôn có:

- Trắng đối diện với Vàng.
- Cam đối diện với Đỏ.
- Lục đối diện với Lam.

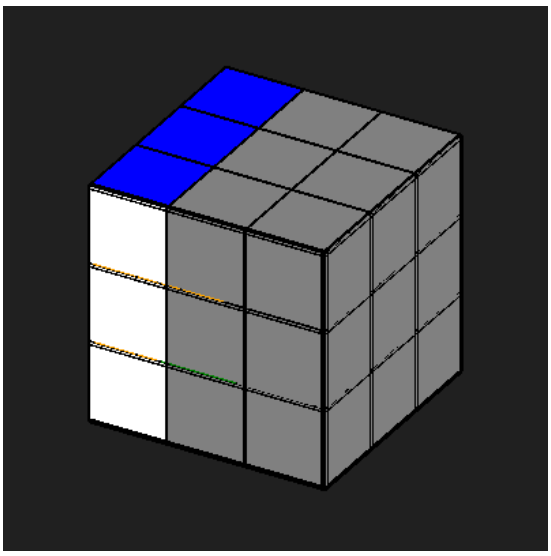
Ký hiệu:

Quy ước tên các mặt của khối Rubik: Chúng ta có 6 mặt, được ký hiệu theo tên viết tắt của các mặt này trong tiếng Anh (để thống nhất các ký hiệu nhằm giúp các bạn có thể xem thêm các cách giải trong nhiều tài liệu khác).

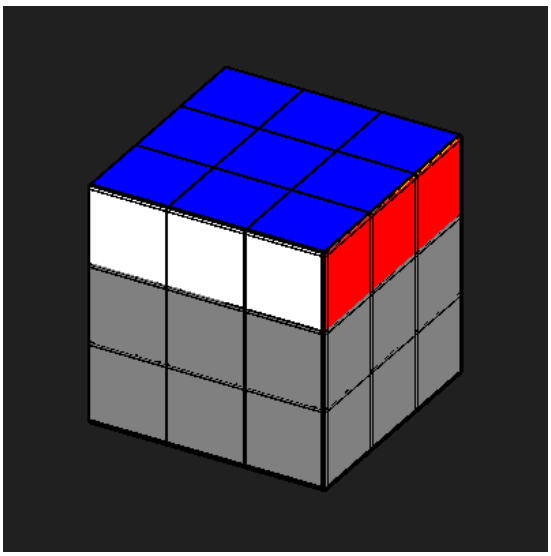
Quy ước các cách xoay của khối Rubik:



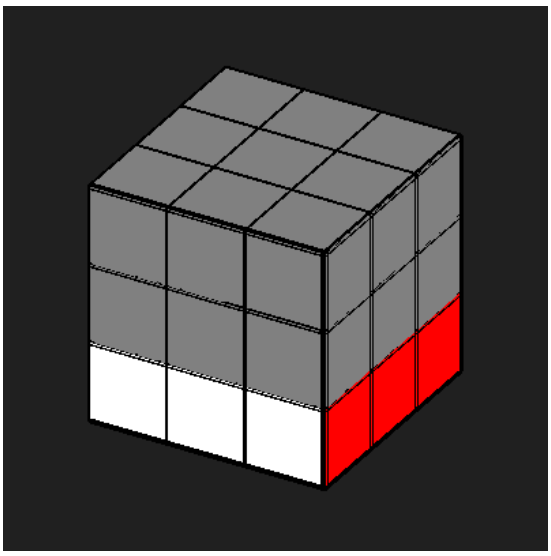
Hình 1-6: mặt phải R của khối Rubik (Right)



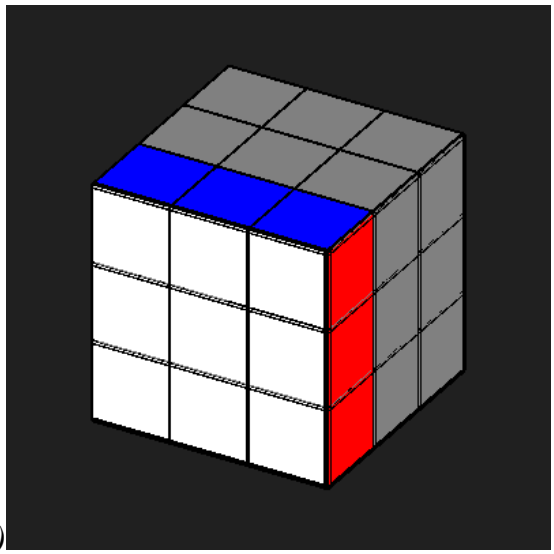
Hình 1-7: mặt trái L của khối Rubik (Left)



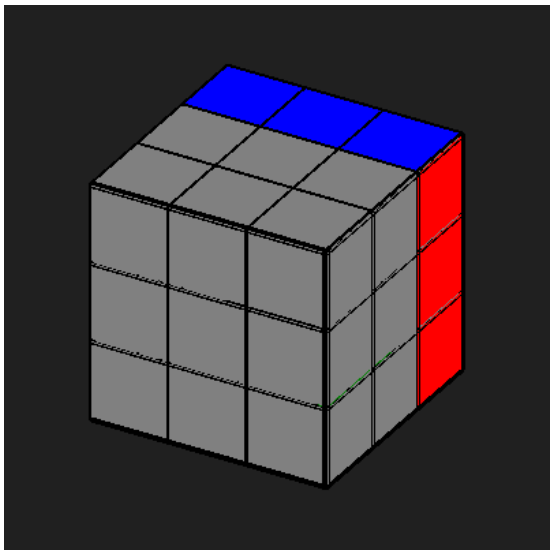
Hình 1-8: mặt trên U của khối Rubik (Up)



Hình 1-9: mặt dưới D của khối Rubik (Down)



Hình 1-10: mặt trước F của khối Rubik (Front)

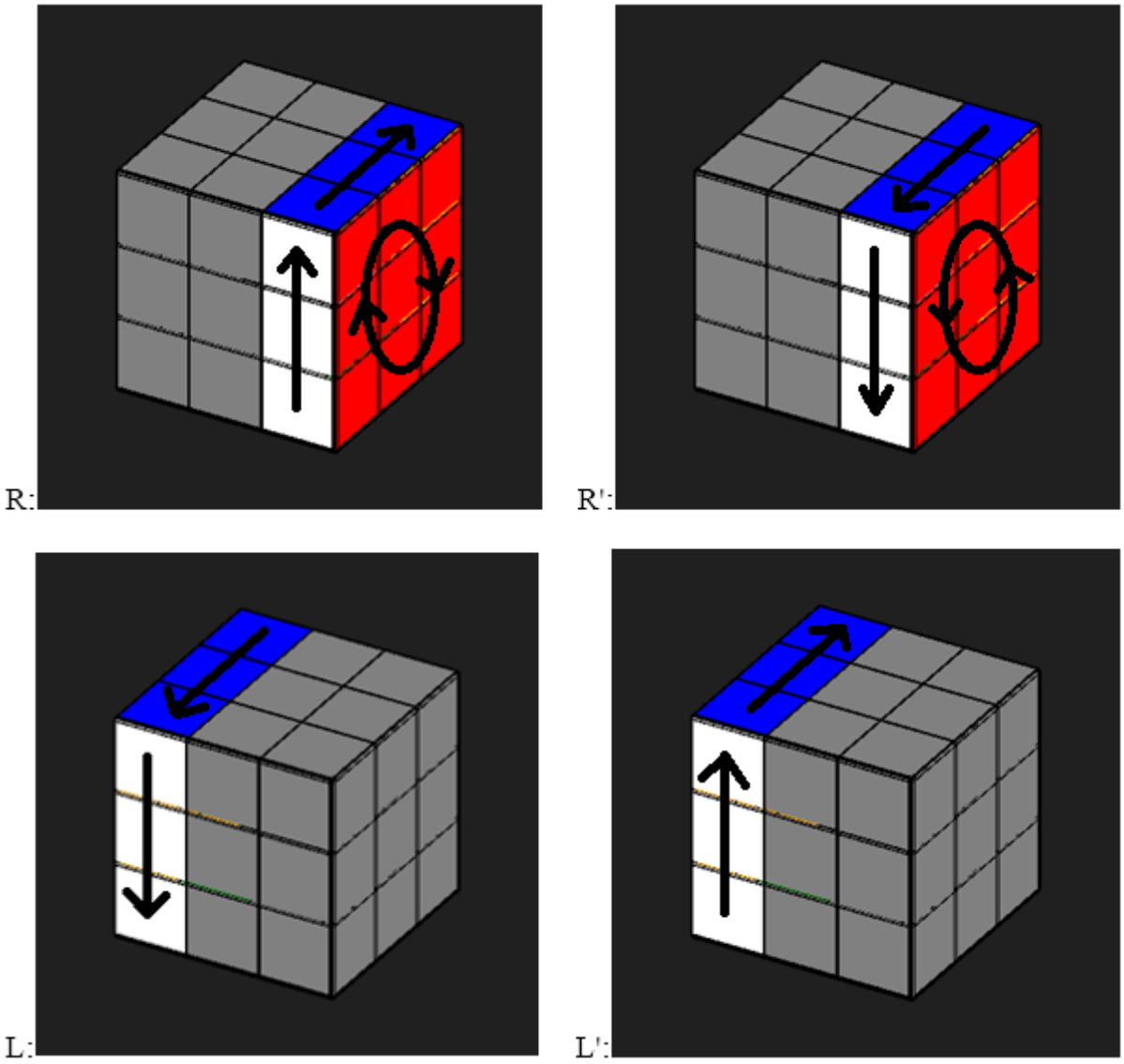


Hình 1-11: mặt sau B của khối Rubik (Back)

Trong công thức, các ký tự R, L, U, D, F, B đại diện cho cách quay Rubik 90 độ theo chiều kim đồng hồ (từ trái qua phải).

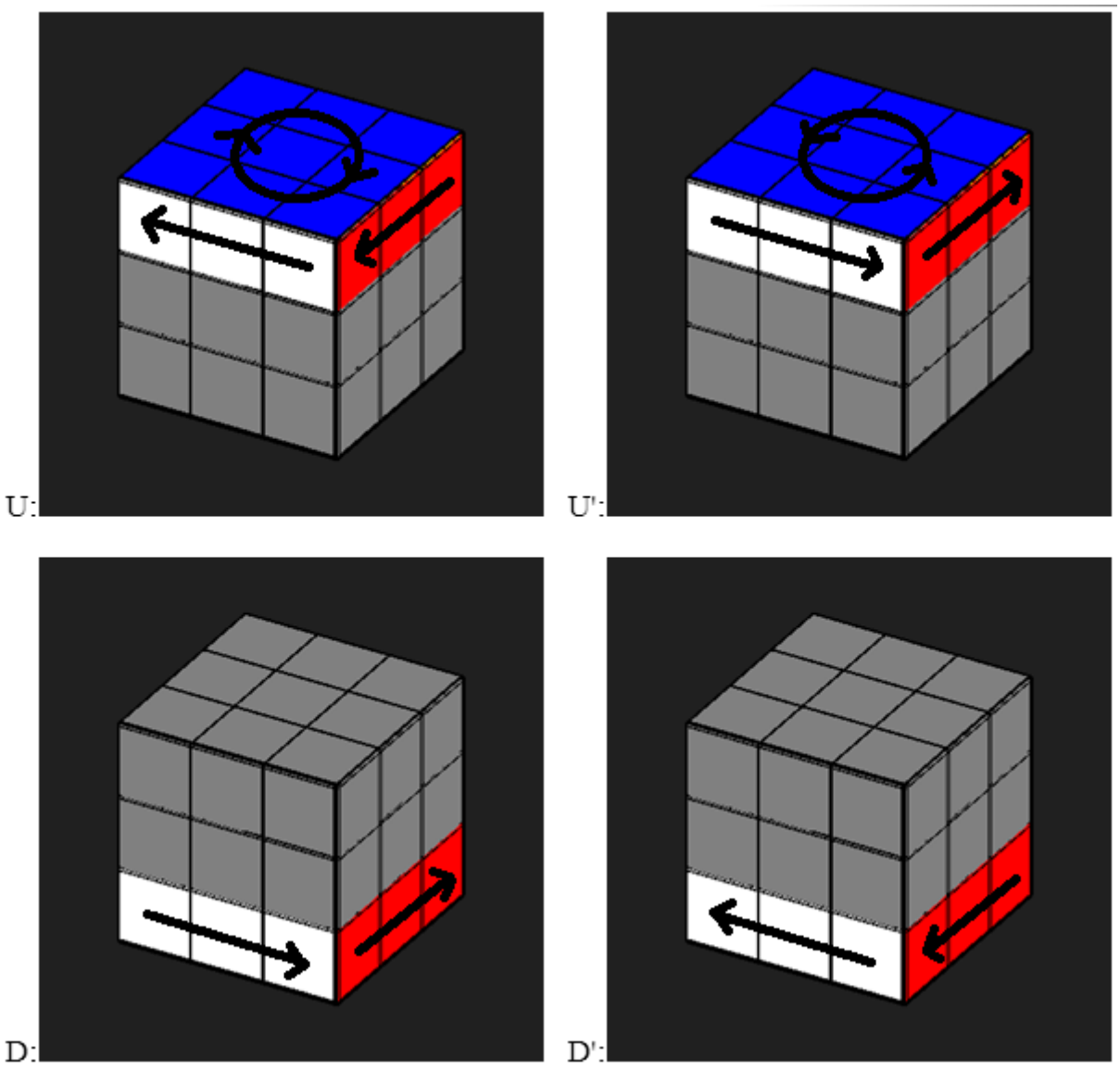
Các ký tự có "i" (Ri, Li, Ui, Di, Fi, Bi) đại diện cho cách quay Rubik 90 độ theo chiều ngược lại (ngược chiều kim đồng hồ, từ phải qua trái).

(Một số tài liệu dùng ký hiệu ' thay vì dùng i: R' L' U' D', F', B'.)

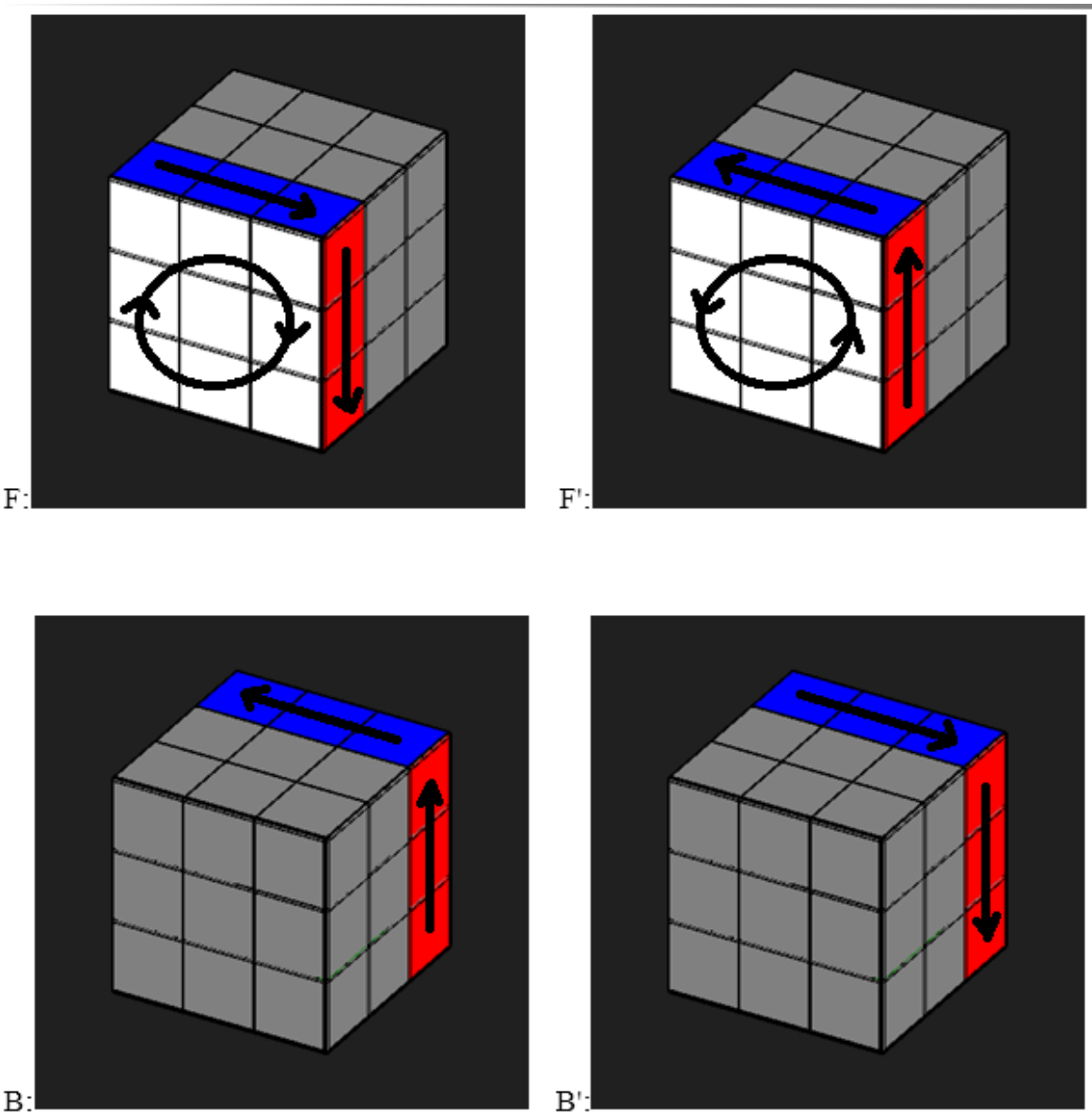


Hình 1-12: Các ký hiệu xoay mặt trái và phải của khối rubik

U



Hình 1-13: Các ký hiệy xoay mặt trên và dưới của khối rubik



Hình 1-14: Các ký hiệu xoay mặt trước, sau của khối rubik

1.4. Các phương pháp để giải khối rubik 3x3

Trong thuật ngữ của người chơi Rubik, một "thuật toán" (khái niệm này khác với thuật toán trong toán học) là một bộ các bước để thực hiện một công việc nào đó: chuyển từ trạng thái ban đầu đến trạng thái mong muốn. Các phương pháp giải khác nhau sử dụng các thuật toán khác nhau; với mỗi thuật toán cần nắm được công dụng và cách dùng.

Hầu hết thuật toán chỉ ảnh hưởng một phần nhỏ của khối mà không thay đổi các phần khác, chẳng hạn như xoay các khối ở góc, đổi vị trí các khối ở cạnh v.v. Một số thuật toán có tác dụng phụ làm thay đổi vị trí các mảnh khác lại thường đòi hỏi ít nước đi hơn và được dùng nhiều khi bắt đầu giải (chưa cần quan tâm tới vị trí các mảnh khác).

Thuật toán xoay nhanh

Các thuật toán xoay nhanh được tạo ra để giải khối Rubik trong thời gian nhanh nhất có thể. Phương pháp thường dùng nhất được phát triển bởi Jessica Fridrich, là phương pháp giải theo từng lớp có kết hợp các bước so với phương pháp thông thường tuy nhiên đòi hỏi người sử dụng phải nhớ một lượng thuật toán khá lớn (118 thuật toán). Các bước chính trong phương pháp này bao gồm: Cross (thực hiện tạo dấu cộng ở mặt chính); F2L (hoàn thành tầng 1 và tầng 2 cùng một lúc); OLL (làm đúng màu mặt cuối cùng); PLL (đổi tất cả các viên còn lại để tạo thành một Cube hoàn chỉnh)

Một phương pháp khác được phát triển bởi Lars Petruss bao gồm việc giải một khối 2×2×2 rồi đến 2×2×3 và các cạnh được giải bởi một bộ thuật toán 3 bước, thường tránh được một thuật toán 32 bước về sau. Do đó phương pháp này được dùng trong các cuộc thi có tính số bước xoay.

Thuật toán căn bản

Hầu hết các phương pháp giải chỉ cần 4 hoặc 5 thuật toán nhưng không hiệu quả, cần tới khoảng 100 hoặc 150 lần xoay để giải, trong khi phương pháp của Fridrich chỉ cần khoảng 55 lần xoay.

Philip Marshall đã phát triển thêm phương pháp của Fridrich, cần 65 lần xoay tuy nhiên chỉ cần nhớ 2 thuật toán.

Phương pháp phát triển bởi Ryan Heise không dạy cho người chơi một thuật toán nhất định mà chỉ ra các quy tắc của khối để người chơi suy luận; phương pháp này có thể giải khối Rubik trong khoảng 40 lần xoay.

Thuật toán tối ưu

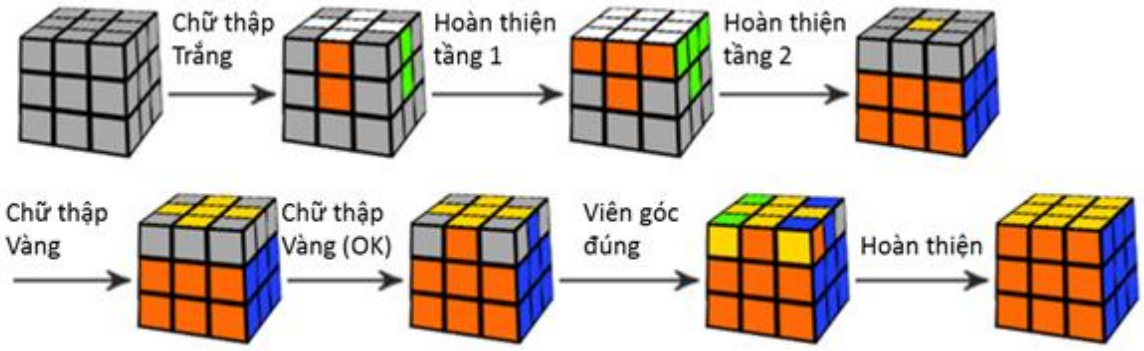
Các thuật toán giải bằng tay đã đề cập tuy dễ học nhưng kém hiệu quả. Từ khi trò chơi được phát minh đã có rất nhiều nỗ lực để tìm các cách giải nhanh hơn.

- Năm 1982, David Singmaster và Alexander Frey đã dự đoán rằng số bước cần thiết để giải khối Rubik là "Khoảng dưới 20"
- Năm 2007 Daniel Kunkle và Gene Cooperman dùng máy tính và các phương pháp tìm kiếm để cho thấy mọi cấu hình của khối 3×3×3 có thể được giải trong 26 bước
- Năm 2008, Tomas Rokicki giảm con số này xuống còn 22 bước
- Năm 2010, Tomas Rokicki, Herbert Kociemba. Morley Davidson và John Dethridge cùng với sự hỗ trợ của các máy chủ của Google đã chứng minh rằng số bước giải cần thiết cao nhất cho bất kì hoán vị nào là 20.

1.5. Giới thiệu phương pháp 7 bước để giải khối rubik 3x3

Với người chơi cơ bản, phương pháp giải Rubik 7 bước được coi là dễ hiểu và dễ thực hiện nhất, giúp bạn dần làm quen được các kí hiệu và công thức.

Để giải khối Rubik 6 mặt 3x3x3 theo phương pháp cơ bản, chúng ta sẽ tiến hành lần lượt thông qua 7 bước với trình tự như sau:



Hình 1-15: 7 bước để hoàn thành khối rubik

Bước 1: Xếp tạo thành hình Chữ thập màu trắng ở tầng 1 của khối Rubik

Bước 2: Hoàn thiện xếp tầng 1 của Rubik

Bước 3: Xoay hoàn thành tầng 2 của khối Rubik

Bước 4: Tạo chữ thập màu vàng ở tầng 3

Bước 5: Đưa các viên chữ thập màu vàng về đúng vị trí

Bước 6: Đưa các viên góc màu vàng về đúng vị trí

Bước 7: Hoàn thiện xếp khối Rubik

Sau khi đã thuần thực 7 bước xoay Rubik cơ bản như trên, bạn có thể chuyển sang Phương pháp quay Rubik nâng cao bằng phương pháp Fridrich với chỉ 4 bước giải để cải thiện kĩ năng và tốc độ quay của mình.

Ngoài ra, trước khi bắt đầu chúng ta cần biết thêm một quy ước mới liên quan tới màu sắc của các tầng của khối Rubik: Tầng 1 được quy ước là tầng có mặt trắng và tầng 3 được quy ước là tầng có mặt màu vàng.

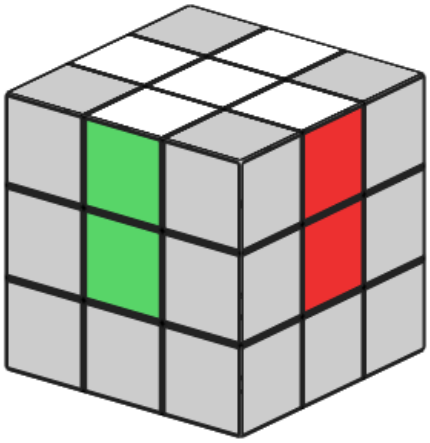
1.6. Giải rubik bằng phương pháp 7 bước

1.6.1. Tạo thành hình Chữ thập màu trắng ở tầng 1

Mục tiêu

Xếp tạo thành chữ thập màu trắng ở tầng 1 của khối Rubik, trong đó các mặt cạnh của các viên màu trắng phải đúng màu với các viên tâm các mặt bên. Để tạo ra chữ thập màu trắng, bạn hoàn toàn có thể giải bằng trực quan. Nhưng nếu không bạn có thể áp dụng các cách thực hiện dưới đây.

Hình 1-16: Bước 1 trong phương pháp 7 bước (1)



Cách thực hiện

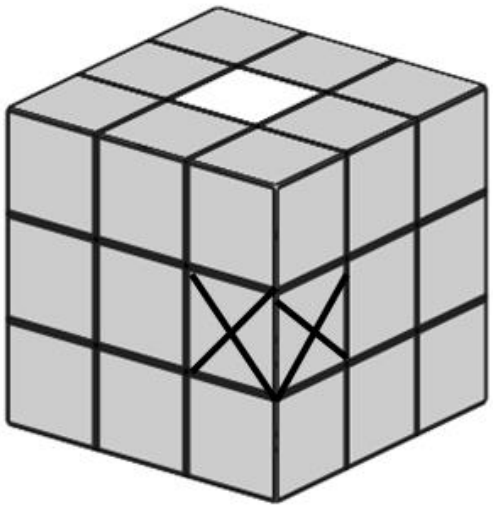
Trước tiên hướng mặt trắng lên phía trên. Để giải một cạnh màu trắng bạn tiến hành các bước sau đây:

Bước 1.1: Xác định 1 viên cạnh có màu trắng cần di chuyển hiện đang ở đâu.

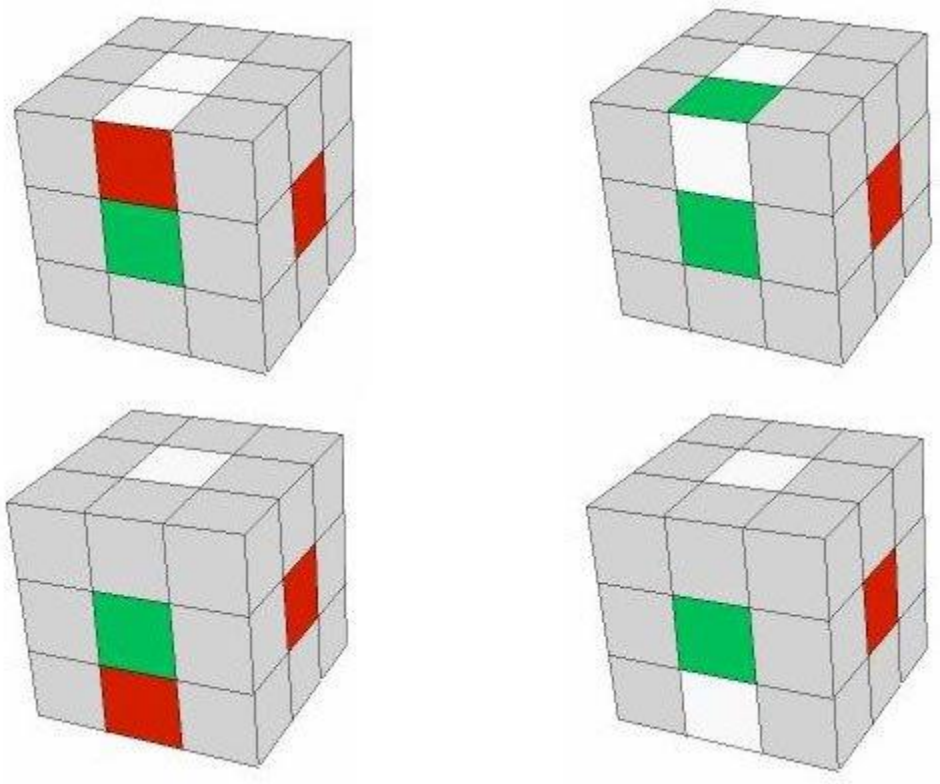
Di chuyển viên cạnh đó ở về mặt trước - F. Lúc này, chúng ta sẽ có ba vị trí của viên cạnh này ở mặt F là: có thể nó ở tầng 1, tầng 2 hoặc tầng 3.

Bằng 1 số phép quay nhất định, đưa nó về vị trí tầng 2, ở mặt trước bên phải, tức vị trí được đánh dấu X dưới hình 1-17.

Hình 1-17: Bước 1 trong phương pháp 7 bước (2)

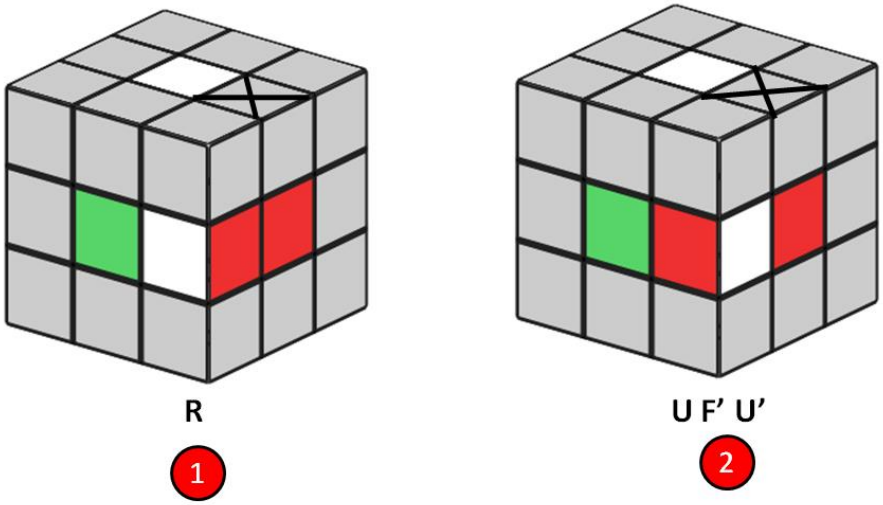


Ví dụ: nếu viên cạnh trắng này đang ở 4 vị trí sau, thì dùng F, F' hoặc F2, F2' để đưa về tầng 2.



Hình 1-18: Bước 1 trong phương pháp 7 bước (3)

Bước 1.2: Xác định vị trí mà viên này sẽ phải trở về. Xoay U (Hoặc U') để đưa vị trí đó về vị trí mặt trên bên phải, tức vị trí X dưới đây. Lúc này, ta sẽ có 2 trường hợp sau:



Hình 1-19: Bước 1 trong phương pháp 7 bước (4)

Bước 1.3: Thực hiện công thức để đưa cạnh về vị trí X

- Trường hợp 1: Đơn giản là xoay R

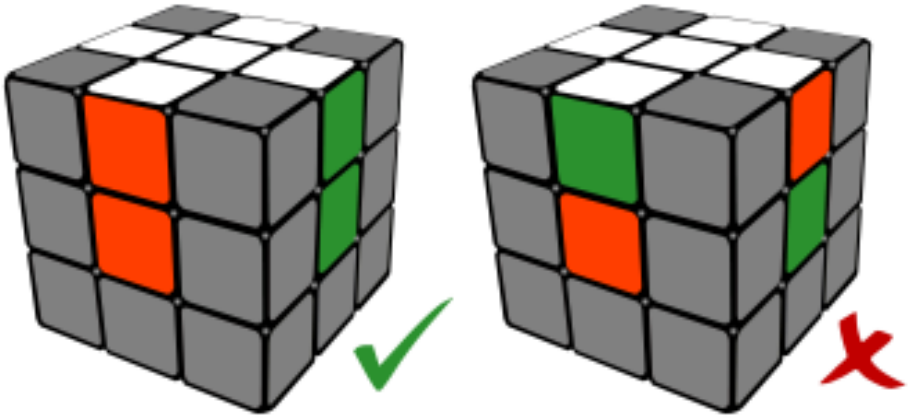
- Trường hợp 2: Thực hiện U F' U' hay U Fi Ui

Hình 1-20: Bước 1 trong phương pháp 7 bước (5)

Lặp lại các bước từ 1.1 đến 1.3 để giải 3 viên cạnh còn lại. Lưu ý, tránh quay các bước quay làm ảnh hưởng tới các cạnh đã giải.

Kết thúc bước 1

Ta được kết quả là một hình chữ thập màu trắng và đúng với các màu trung tâm như sau:



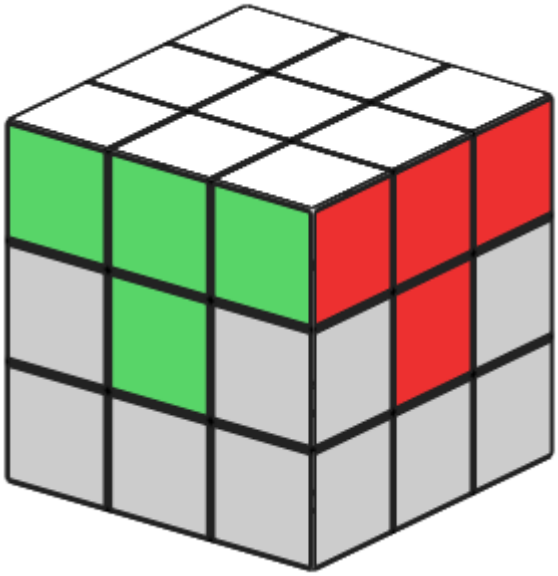
Hình 1-21: Bước 1 trong phương pháp 7 bước (6)

1.6.2. Hoàn thiện tầng 1 của Rubik

Mục tiêu

Giải tất cả các viên góc màu trắng để hoàn thiện tầng 1.

Hình 1-22: Bước 2 trong phương pháp 7 bước (1)

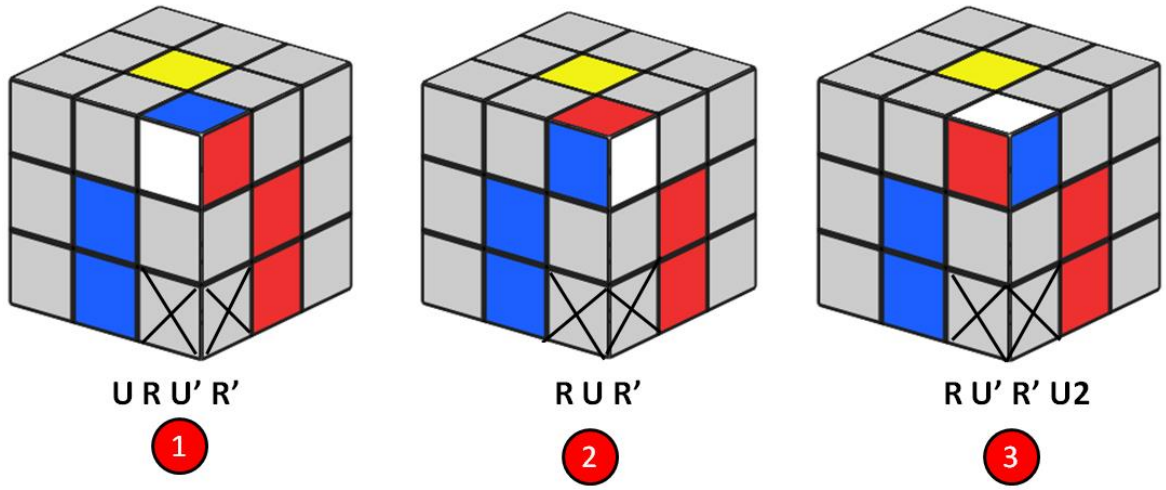


Cách thực hiện

Để thuận tiện, ta quay ngược khối Rubik lại, có nghĩa là mặt màu trắng sẽ là ở dưới, mặt màu vàng trở thành trên.

Với khối Rubik này, bạn hãy quan sát tất cả khối một lượt trước khi đọc tiếp phần hướng dẫn, để xác định vị trí của các ô góc đang ở đâu. Ô góc màu trắng là ô có 3 màu, 1 mặt là màu trắng. Ở đây X được đánh dấu là vị trí mà ô góc đó phải trở về.

- Nếu viên góc nằm ở tầng 3 (tức tầng màu vàng), dùng U hoặc U' để đưa về 3 trường hợp sau:



Hình 1-23: Bước 2 trong phương pháp 7 bước (2)

+ Với hình 1**:** Bạn sử dụng công thức xoay U R U' R'

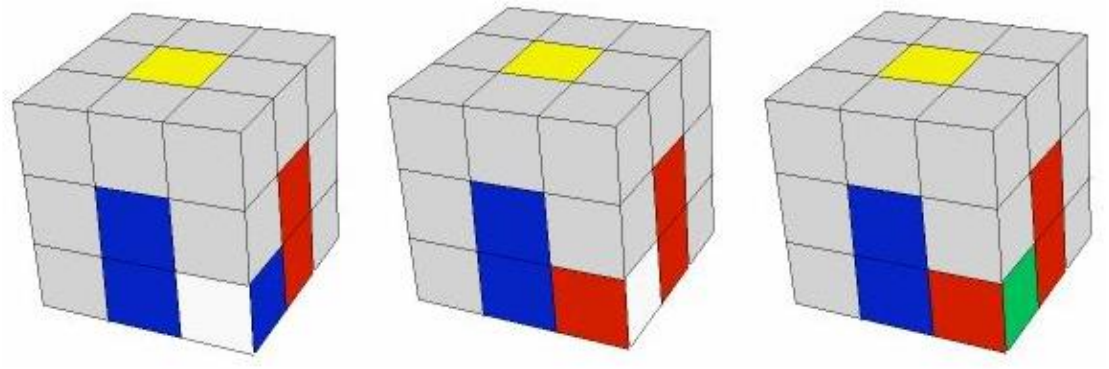
+ Với hình 2**:** Bạn sử dụng công thức xoay R U R'

+ Với hình 3: Vị trí của mặt viên góc khác một chút so với hình 1 và 2, đó là mặt màu trắng không ở mặt cạnh (xanh, đỏ) mà ở mặt màu vàng. Do đó **đ**ầu tiên, đưa mặt viên màu trắng này sang bên cạnh như hình 1 và 2 bằng cách xoay R U' R' U2 .

Tiếp theo: chọn một trong hai công thức hình 1 hoặc hình 2 để giải tiếp.

- Nếu viên góc ở tầng 1 (tức ở tầng màu trắng)

Ở trường hợp này, ta thấy rằng viên góc này đang ở đúng tầng 1, nhưng đang sai vị trí hoặc sai hướng. Có 3 trường hợp như sau:

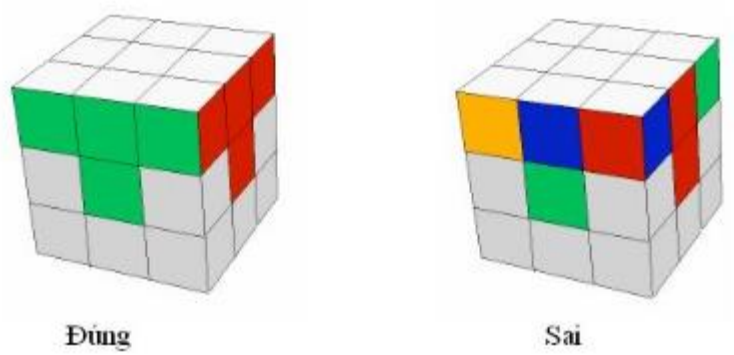


Hình 1-24: Bước 2 trong phương pháp 7 bước (3)

Để giải, trước tiên, dùng công thức (R U R’ U’) để đưa viên góc về tầng 3. Sau đó, dùng phương pháp giải tầng 3 như trên để giải tiếp.

Kết quả sau bước 2

Tầng 1 hoàn thành đồng thời các ô cạnh đúng với vị trí màu các bên như hình.



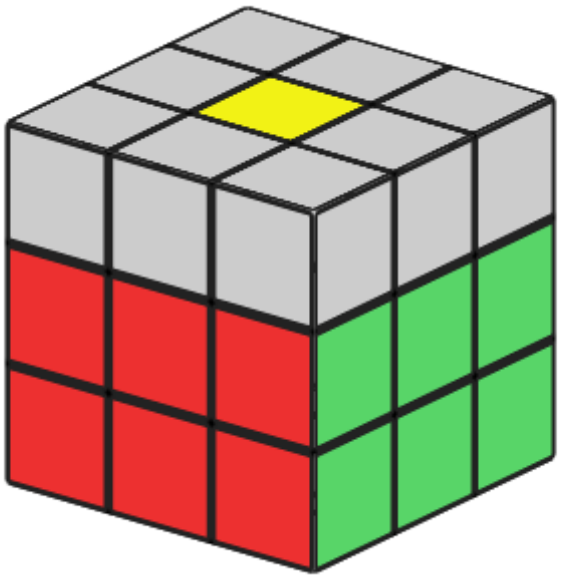
Hình 1-25: Bước 2 trong phương pháp 7 bước (4)

1.6.3. Hoàn thành tầng 2 của khối Rubik

Mục tiêu

Ở tầng 2, công việc rất nhẹ nhàng, ta chỉ cần giải 4 viên cạnh, đưa chúng về đúng vị trí ở tầng 2.

Hình 1-26: Bước 3 trong phương pháp 7 bước (1)



Cách thực hiện

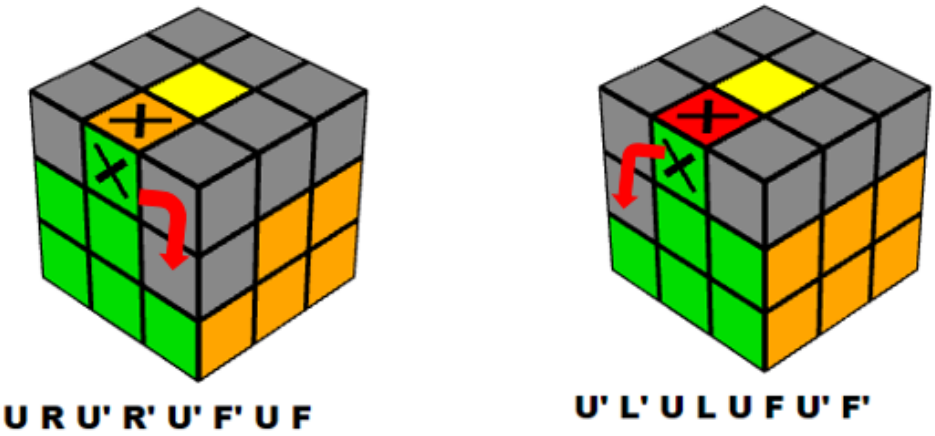
Đầu tiên ta xác định các viên cạnh của tầng 2, đó là các viên cạnh còn lại mà không có màu vàng. Các viên này có thể nằm ở tầng 2 hoặc tầng 3.

- Nếu viên cạnh nằm ở tầng 3

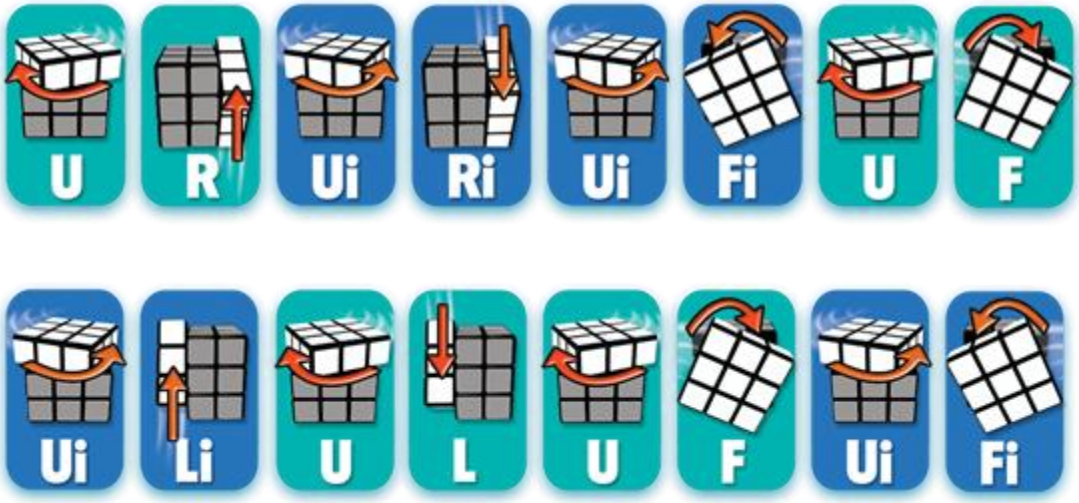
Bước 1: Xác định vị trí viên cạnh cần đưa tới bằng cách xem xét 2 màu của viên cạnh. Ta gọi vị trí cần tới đó là Goal. Cầm Rubik sao cho viên Goal nằm ở mặt F.

Bước 2: Xoay U, U' hoặc U2 để đưa viên cạnh đến vị trí gần Goal sao cho trục giữa của mặt F trùng màu, tạo thành chữ T (xem hình minh họa phía dưới).

Bước 3**:** Tùy vào từng trường hợp, dùng 1 trong 2 công thức sau để giải:



Hình 1-27 Bước 3 trong phương pháp 7 bước (2)



Hình 1-28: Bước 3 trong phương pháp 7 bước (3)

- Nếu viên cạnh nằm ở tầng 2

Bước 1**:** Dùng công thức (R U’ R’) (U’ F’ U F) để xếp và xoay viên cạnh về tầng 3.

Bước 2**:** Dùng phương pháp phía trên để giải.

1.6.4. Tạo chữ thập màu vàng ở tầng 3

Cuối cùng là tầng 3, tầng này luôn luôn khó khăn nhất, nếu bạn làm sai 1 bước nhỏ có thể dẫn đến chúng ta phải bắt đầu lại. Đây thực sự là tầng có phần giải khá phức tạp bạn cần thật tỉ mỉ và kiên nhẫn theo từng bước hướng dẫn:

Mục tiêu

Tạo thành hình chữ thập màu vàng ở tầng 3 của Rubik nhưng không cần phải đúng màu với các cạnh.

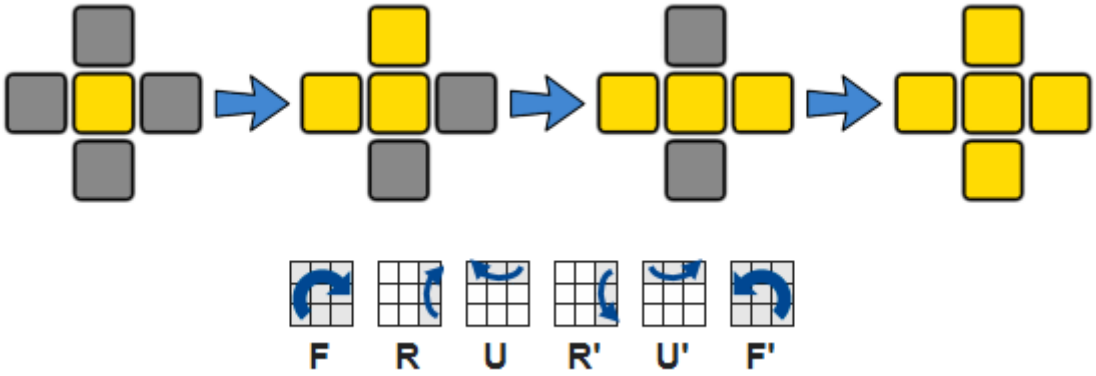
Cách thực hiện

Cách 1: Ở bước này, mặc dù chúng ta có 3 trường hợp của tầng 3 là: chỉ có 1 Dot, có 3 Dot hình chữ L, có 3 Dot theo đường thẳng. Nhưng phương pháp ở bước này chỉ cần một công thức đó là**:** F R U R' U' F'**.**

- Trong trường hợp 1 Dot: chúng ta cần xoay công thức này ba lần

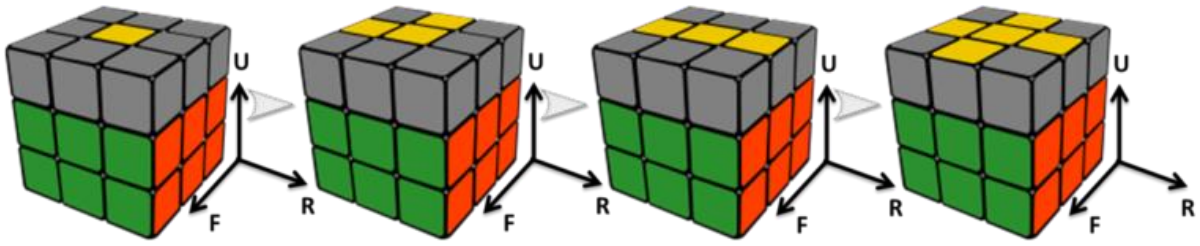
- Trong trường hợp 3 Dot chữ L: chúng ta cần xoay hai lần. Lưu ý hướng của chữ L.

- Trong trường hợp 3 Dot đường thẳng nằm ngang: chúng ta xoay công thức này 1 lần



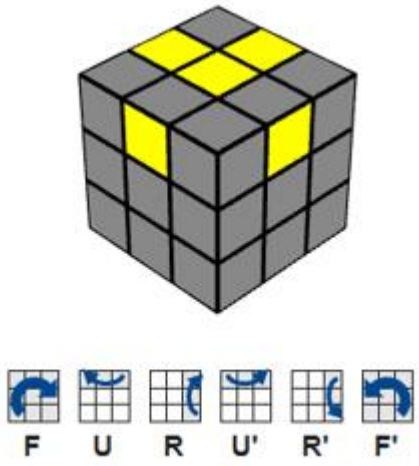
Hình 1-29: Bước 4 trong phương pháp 7 bước (1)

Khi đó tầng 3 của khối Rubik sẽ lần lượt thay đổi theo thứ tự như hình dưới. Lưu ý, Hướng của khối Rubik rất quan trọng, vì vậy hình dạng "L" phải có dạng như minh họa và đường thẳng phải nằm ngang.



Hình 1-30: Bước 4 trong phương pháp 7 bước (2)

Cách 2: Đây là 1 cách làm tắt, nếu như bạn đang ở dạng chữ L, bạn sẽ có thể chuyển ngay đến dạng chữ thập mà chỉ cần xoay 1 lần công thức là: F U R U' R 'F'



Hình 1-31: Bước 4 trong phương pháp 7 bước (4)

1.6.5. Đưa các viên chữ thập màu vàng về đúng vị trí

Mục tiêu

Sau bước 4, chúng ta đã tạo ra được chữ thập màu vàng ở tầng 3, nhưng có thể vị trí của chúng không đúng. Vì vậy bước này giúp đưa lại chúng về đúng vị trí, tức là các các mặt cạnh trùng với màu của viên tâm.

Cách thực hiện

Quan sát khối Rubik, kiểm tra vị trí của các mảnh cạnh màu vàng cần chuyển đổi. Cầm Rubik sao cho hai cạnh cần hoán đổi với nhau nằm ở mặt trước F và mặt trái L.

Thực hiện công thức **(**R U**) (**R’ U**) (**R U2**)** R’ U để hoán vị giữa cạnh vàng mặt F với cạnh vàng mặt L.

Hình 1-32: Bước 5 trong phương pháp 7 bước (1)

Kết quả bước 5

Chúng ta sẽ được hình khối rubik như sau:

Hình 1-33: Bước 5 trong phương pháp 7 bước (2)

1.6.6. Đưa các viên góc màu vàng về đúng vị trí

Mục tiêu

Đưa các viên góc màu vàng về đúng vị trí của chúng (nhưng có thể sai hướng)

Cách thực hiện

Quan sát xem các viên góc màu vàng có viên nào đang nằm sai vị trí không. Đúng vị trí được hiểu là viên có 3 góc màu: màu vàng và 2 màu còn lại đang nằm ở giao điểm tại 3 cạnh có màu tương ứng (không nhất thiết trùng màu tâm).

Ở bước này, một điều thú vị chúng ta sẽ thấy rằng: sẽ luôn chỉ có 0, 1 hoặc là 4 viên góc ở vị trí đúng.

- Nếu có 1 viên góc ở vị trí đúng: Cầm khối Rubik sao cho viên đúng này ở vị trí FRU (Mặt trước, phía trên, bên trái). Áp dụng công thức: U R U’ L’ U R’ U’ L

Hình 1-34: Bước 6 trong phương pháp 7 bước (1)

- Nếu không có viên góc nào ở vị trí đúng: bạn cần thực hiện công thức trên khoảng 2 lần để tạo được 1 góc đúng.

- Nếu cả 4 viên góc đúng thì chúc mừng bạn, bạn có thể chuyển ngay tới bước 7

1.6.7. Hoàn thành giải khối Rubik

Mục tiêu

Hoàn thiện giải khối Rubik bằng cách hoán đổi hướng đúng của các ô góc ở bước 6 nếu như chúng chưa đúng hướng.

Cách thực hiện

Sau bước 6, nếu các viên góc vô tình quay đúng hướng thì chúc mừng bạn, bạn đã hoàn thành khối Rubik mà không cần đến bước 7. Còn nếu không bạn cần thực hiện như sau:

- Chọn hướng cầm Rubik sao cho 1 viên góc màu vàng bị sai hướng nằm ở mặt trước, phía trên, bên phải như vị trí đánh dấu như bên dưới, tức vị trí FRU.

Hình 1-35: Bước 5 trong phương pháp 7 bước (1)

- Thực hiện chẵn lần (2 hoặc 4 lần) công thức sau**:** R’ D’ R D để định hướng đúng góc này, vì khi thực hiện công thức này, mặt màu vàng sẽ xoay tại chỗ theo chiều kim đồng hồ. Dừng thực hiện khi mặt vàng ở đúng vị trí. Việc xáo trộn các tầng 1, 2 không có vấn đề gì cả vì chúng sẽ tự về vị trí đúng sau khi bạn giải xong tất cả các góc sai.

Hình 1-36: Bước 5 trong phương pháp 7 bước (2)

- Dùng U / U' để chuyển các ô vàng sai hướng còn lại đến vị trí đánh dấu FRU và tiếp tục áp dụng lại công thức trên cho đến khi tất cả các ô góc vàng được giải.

Lưu ý: Ngoại trừ viên góc đầu tiên, chỉ sử dụng U và U' để di chuyển các góc còn lại tới vị trí FRU.

Ví dụ: Có 2 góc cần định hướng và liền nhau

(R' D' R D)x2 U' (R' D' R D)x4 U

(R' D' R D)x4 U' (R' D' R D)x2 U

Hình 1-37: Bước 5 trong phương pháp 7 bước (3)

Kết thúc : Khi tất cả các góc được định hướng chuẩn, thì bạn đã giải xong khối Rubik! Chúc mừng bạn đã giải được khối Rubik Cube 3x3x3.

Hình 1-38: Bước 5 trong phương pháp 7 bước (4)

1.7. Ý tưởng, mục tiêu của chương trình Hướng dẫn chơi rubik

Nhận thấy được thực tế là có rất nhiều người gặp khó khăn trong việc bắt đầu tập chơi rubik. Những hạn chế chúng ta thấy rõ được như người chơi không có sẵn rubik, những hướng dẫn trên Internet quá khô khan, có những trường hợp khó của rubik thì ít khi xảy ra nên người chơi không chủ động được trong việc thực hành tập xoay những trường hợp khó.

Do đó, nhóm em sẽ hiện thực chương trình Hướng dẫn chơi rubik hướng đến đối tượng người chơi mới bắt đầu tập chơi hay muốn rèn luyện khả năng chơi rubik của bản thân.

Mục tiêu của chương trình sẽ là:

- Giao diện đẹp, thân thiện với người dùng.
- Dễ dàng sử dụng, phù hợp với mọi lứa tuổi.
- Đầy đủ các chức năng giúp người chơi có thể vừa học vừa chơi.

Chương trình mà nhóm em hiện thực sẽ chỉ hướng đến việc hướng dẫn chơi rubik 3x3. Sau đây khi nhắc đến rubik trong báo cáo luận văn thì nhóm em muốn ám chỉ đến khối rubik 3x3.

2. GIẢI PHÁP THỰC HIỆN

2.1. Ngôn ngữ hiện thực: C# (Winform)

Winform là thuật ngữ mô tả một ứng dụng được viết dùng .NET FrameWorrk và có giao diện người dùng Windows Forms.

Mỗi màn hình windows cung cấp một giao diện giúp người dùng giao tiếp với ứng dụng. Giao diện này được gọi là giao diện đồ họa (*GUI*) của ứng dụng.

Là các ứng dụng windows chạy trên máy tính – mã lệnh thực thi ngay trên máy tính: Microsoft, Word, Excel, Access, Calculator, yahoo, Mail… là các ứng dụng **Windows Forms**.

Ưu điểm:

Đa phần lập trình viên C#. NET nào cũng từng học/sử dụng Winform. Bởi vì: Giao diện kéo thả dễ sử dụng; Gắn các event cho các button chỉ cần double click, lại hỗ trợ quá trời event như click, hover,…; Việc viết code cũng vô cùng trực quan: từ việc lấy text từ TextBox cho tới show dữ liệu bằng MessageBox, hoặc dùng Grid để kết nối SQL. **WinForm rất dễ học và dễ dạy**.

Vì dễ code, chỉ cần kéo thả, lại có nhiều component có sẵn, WinForm rất phù hợp để làm các phần mềm quản lý, tính tiền, thống kê… . Đây cũng là loại ứng dụng mà các công ty/doanh nghiệp vừa và nhỏ cần. Ngoài ra, chỉ cần sử dụng component như TelerikUI hoặc DevExpress, WinForm có thể tạo ra các giao diện hiện đại, đẹp.

- Tốc độ xử lý dữ liệu nhanh chóng
- Đảm bảo an toàn, bảo mật thông tin
- Có thể chạy trên các phiên bản Windows khác nhau.
- Thao tác trên nhiều giao diện

Nhược điểm:

- Phần mềm chạy trên nền tảng Windows đó chính là người dùng muốn dùng phần mềm sẽ phải sử dụng máy tính đã cài phần mềm. Do vậy, bạn phải mang theo mình chiếc máy tính cá nhân để phục vụ cho công việc.
- Winform chỉ phù hợp các ứng dụng trên desktop: ứng dụng quản lý thông tin, ứng dụng tương tác trực tiếp với người dùng.

- Đồ họa trên winform không cao nên giao diện phần mềm sẽ thiếu tính trực quan, hơi khó thao tác, không thân thiện với người dùng.

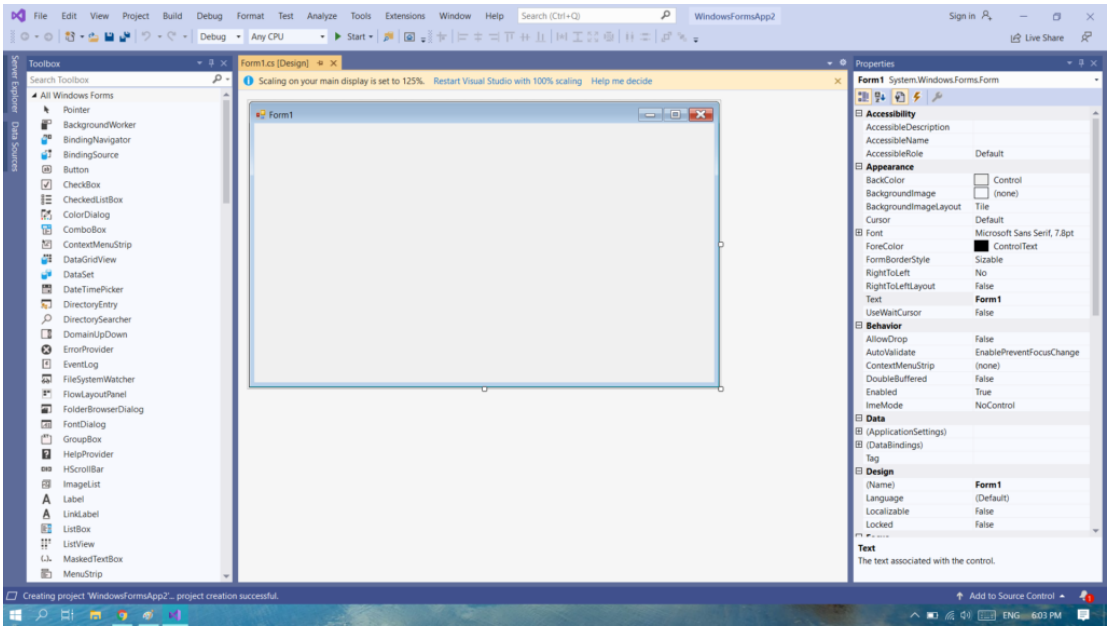
2.2. IDE được sử dụng: Visual Studio

Visual studio là một trong những công cụ hỗ trợ lập trình website rất nổi tiếng nhất hiện nay của Mcrosoft và chưa có một phần mềm nào có thể thay thế được nó. Visual Studio được viết bằng 2 ngôn ngữ đó chính là C# và VB+. Đây là 2 ngôn ngữ lập trình giúp người dùng có thể lập trình được hệ thống một các dễ dàng và nhanh chóng nhất thông qua Visual Studio.

Visual Studio là một phần mềm lập trình hệ thống được sản xuất trực tiếp từ Microsoft. Từ khi ra đời đến nay, Visual Studio đã có rất nhiều các phiên bản sử dụng khác nhau. Điều đó, giúp cho người dùng có thể lựa chọn được phiên bản tương thích với dòng máy của mình cũng như cấu hình sử dụng phù hợp nhất.

Bên cạnh đó, Visual Studio còn cho phép người dùng có thể tự chọn lựa giao diện chính cho máy của mình tùy thuộc vào nhu cầu sử dụng.

Tại sao lại là Visual Studio? Thay vì phải dùng bộ soạn mã để tạo ra giao diện cho một ứng dụng windows thì IDE Visual Sudio .NET hỗ trợ cho chúng ta một bộ công cụ kéo thả, giúp việc thiết kế giao diện trở nên dễ dàng và nhanh chóng hơn bao giờ hết.



Hình 2-1: Giao diện Visual Studio

Như đã nói ở trên bộ IDE Visual Studio tích hợp cho chúng ta cả bộ công cụ kéo thả như trong hình, việc của chúng ta là kéo và kéo nó qua màn hình làm việc, sáng tạo thiết kế một giao diện cho riêng mình.

Hộp thoại Toolbox nằm bên trái cung cấp cho chúng ta tất cả các control có thể có trong giao diện như các dòng text, hộp thoại textbox, hay các button…

Hộp thoại Properties nằm bên phải dùng để điều chỉnh các thuộc tính của một đối tượng nằm trên màn hình Windows Forms.

Và màn hình chính giữa chính là giao diện của một ứng dụng windows, ở đây chúng ta có thể dễ dàng thiết kế giao diện cho một application.

2.3. Thư viện chủ yếu: OpenTK, OpenTK.GLControl, OpenCV

Thư viện OpenTK được sử dụng trong chương trình này chủ yếu để sử dụng thư viện OpenGL được bao gồm bên trong bằng ngôn ngữ C#. Phiên bản được sử dụng để hiện thực chương trình là OpenTK 3.0.0 và OpenTK.GLControl 3.0.0.

Về OpenGL:

OpenGL chỉ định một tập hợp các "lệnh" hoặc các hàm phải được thực thi ngay lập tức. Trong đó mỗi lệnh phụ trách một hành động vẽ hoặc tạo ra các hiệu ứng đặc biệt nào đó. Một danh sách các lệnh như vậy có thể được tạo ra để tạo các hiệu ứng lặp đi lặp lại OpenGL độc lập với các đặc tính của mỗi hệ điều hành, nhưng cung cấp các quy trình "glue" đặc biệt cho mỗi hệ điều hành, điều này cho phép OpenGL hoạt động được trong môi trường của hệ thống đó.

OpenGL chứa đựng một số lượng lớn các tính năng tích hợp được chỉ định và yêu cầu thông qua API, bao gồm loại bỏ bề mặt ẩn, trộn alpha, chống hiệu ứng răng cưa, làm mịn, tính toán pixel, theo dõi và biến đổi các mô hình, và các hiệu ứng về không khí (sương mù, khói và khói mù).

Silicon Graphics, nhà sản xuất máy trạm đồ họa tiên tiến thế giới, là đơn vị đi tiên phong trong việc phát triển OpenGL. Theo sau là các công ty khác trong Architecture Review Board bao gồm DEC, Intel, IBM, Microsoft và Sun Microsystems. Không tốn bất cứ chi phí nào (ngoài việc học) cho việc phát triển một ứng dụng sử dụng API OpenGL. Ngoài ra Microsoft cung cấp các thư viện OpenGL cho phép người dùng tải miễn phí trên các hệ thống windows của hãng.

Tóm lại, OpenGL được thiết kế nhằm thỏa mãn mục đích chính sau:

- Đơn giản hóa việc tương tác giữa các mô hình không gian 3 chiều bằng một giao diện lập trình thống nhất.
- Hỗ trợ tối đa các chức năng của giao diện OpenGL bằng cách ép buộc các phần cứng 3 chiều khác nhau phải tương thích. Ngay cả khi không thể ép phần cứng hỗ trợ hoàn toàn, OpenGL có thể yêu cầu hệ thống sử dụng thêm sức mạnh phần mềm để xử lý.
- Tiêu chuẩn OpenGL nhận các nguyên hàm hình học như điểm, đường thẳng và đa giác rồi chuyển thành các điểm đồ họa (pixel) trên màn hình. Quá trình này được thực hiện thông qua luồng ống dẫn đồ họa (thuật ngữ graphics pipeline). Một tên gọi khác của OpenGL cũng được giới kỹ thuật chia sẻ đó là bộ máy trạng thái OpenGL.

Về OpenCV:

OpenCV là tên viết tắt của open source computer vision library – có thể được hiểu là một thư viện nguồn mở cho máy tính. Cụ thể hơn OpenCV là kho lưu trữ các mã nguồn mở được dùng để xử lý hình ảnh, phát triển các ứng dụng đồ họa trong thời gian thực.

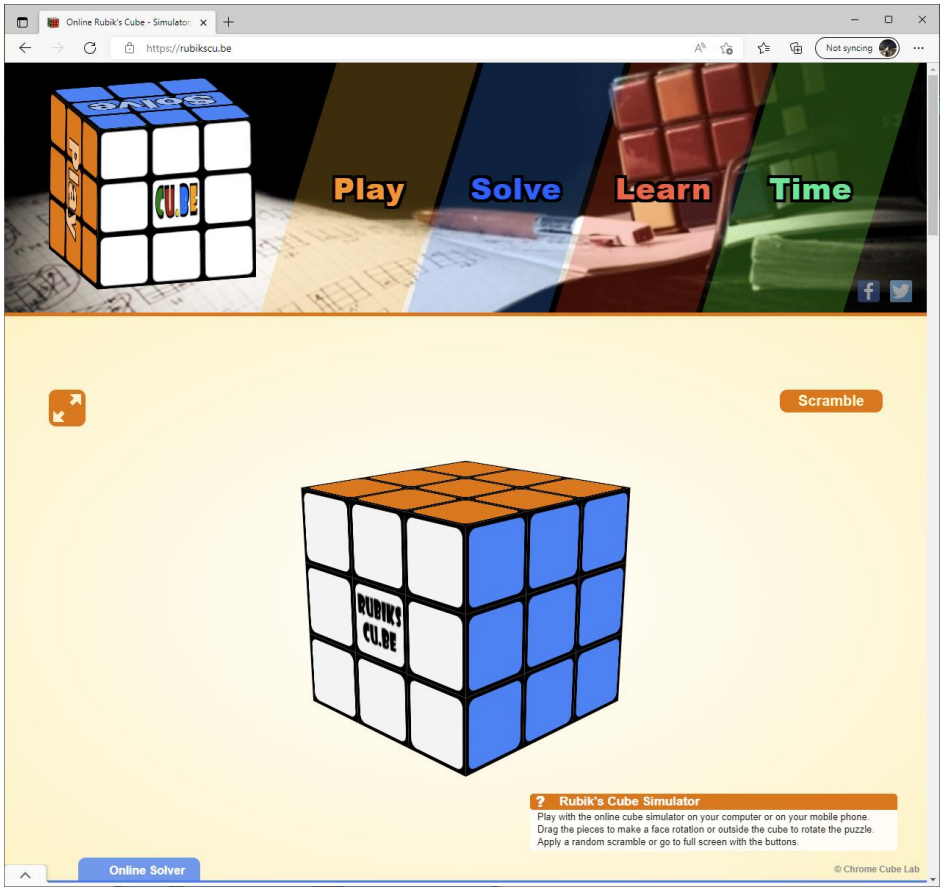
OpenCV cho phép cải thiện tốc độ của CPU khi thực hiện các hoạt động real time. Nó còn cung cấp một số lượng lớn các mã xử lý phục vụ cho quy trình của thị giác máy tính hay các learning machine khác.

Thư viện OpenCV được phát hành với giấy phép BDS. Do đó các dịch vụ nó cung cấp là hoàn toàn miễn phí và được hạn chế tối đa các rào cản thông thường. Cụ thể, bạn được phép sử dụng phần mềm này cho cả hoạt động thương mại lẫn phi thương mại. OpenCV sở hữu giao diện thiên thiện với mọi loại ngôn ngữ lập trình, ví dụ như C++, C, Python hay Java… Ngoài ra, nó cũng dễ dàng tương thích với các hệ điều hành khác nhau, bao gồm từ Windows, Linux, Mac OS, iOS cho đến cả Android.

3. THIẾT KẾ CHƯƠNG TRÌNH VÀ MỘT SỐ THUẬT TOÁN PHỨC TẠP

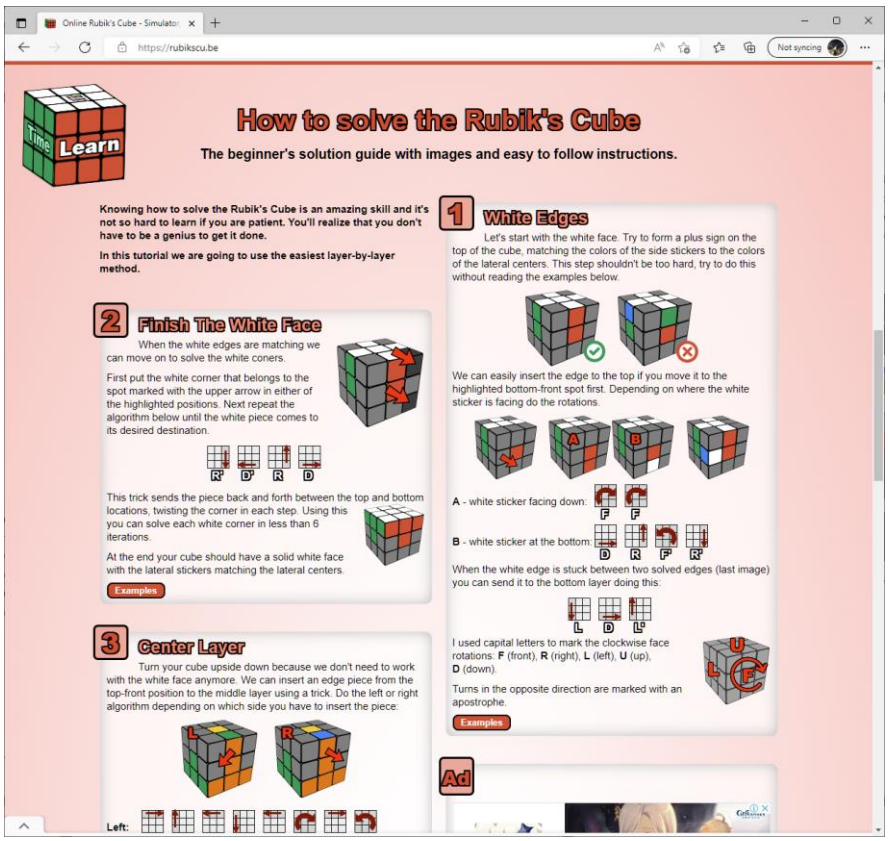
3.1. Tham khảo các chương trình liên quan

3.1.1. Rubikscu.be



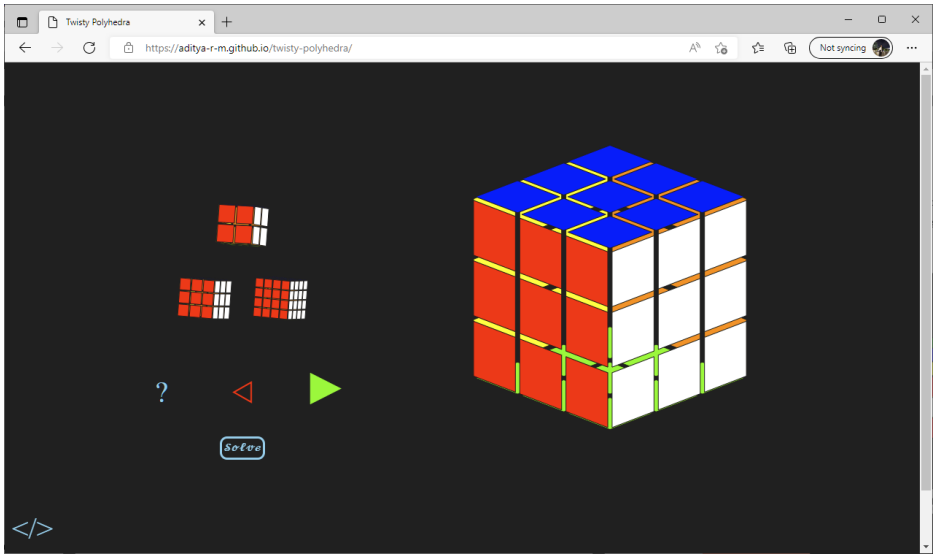
Hình 3-1: Giao diện chính của rubikscu.be

Rubikscu.be là trang web hỗ trợ người chơi rubik với 4 chức năng chính là Play, Solve, Learn và Time. Hình 3-1 là giao diện chính của trang web ở chức năng Play với khối rubik giả lập được thiết kế đẹp mắt. Tuy nhiên nhóm nhận thấy những hạn chế của trang web như chức năng solve không thể đưa ra lời giải sử dụng phương pháp 7 bước như trong chức năng Learn hướng dẫn, cách nhập dữ liệu khối rubik còn thủ công, chức năng Learn chỉ gồm lí thuyết nên người chơi vẫn còn khó khăn trong tiếp cận giải khối rubik.



Hình 3-2: Giao diện chức năng Learn của trang web rubikscu.be

3.1.2. Twisty Polyhedra by Aditya



Hình 3-3: Giao diện chính của Twisty Polyhedra

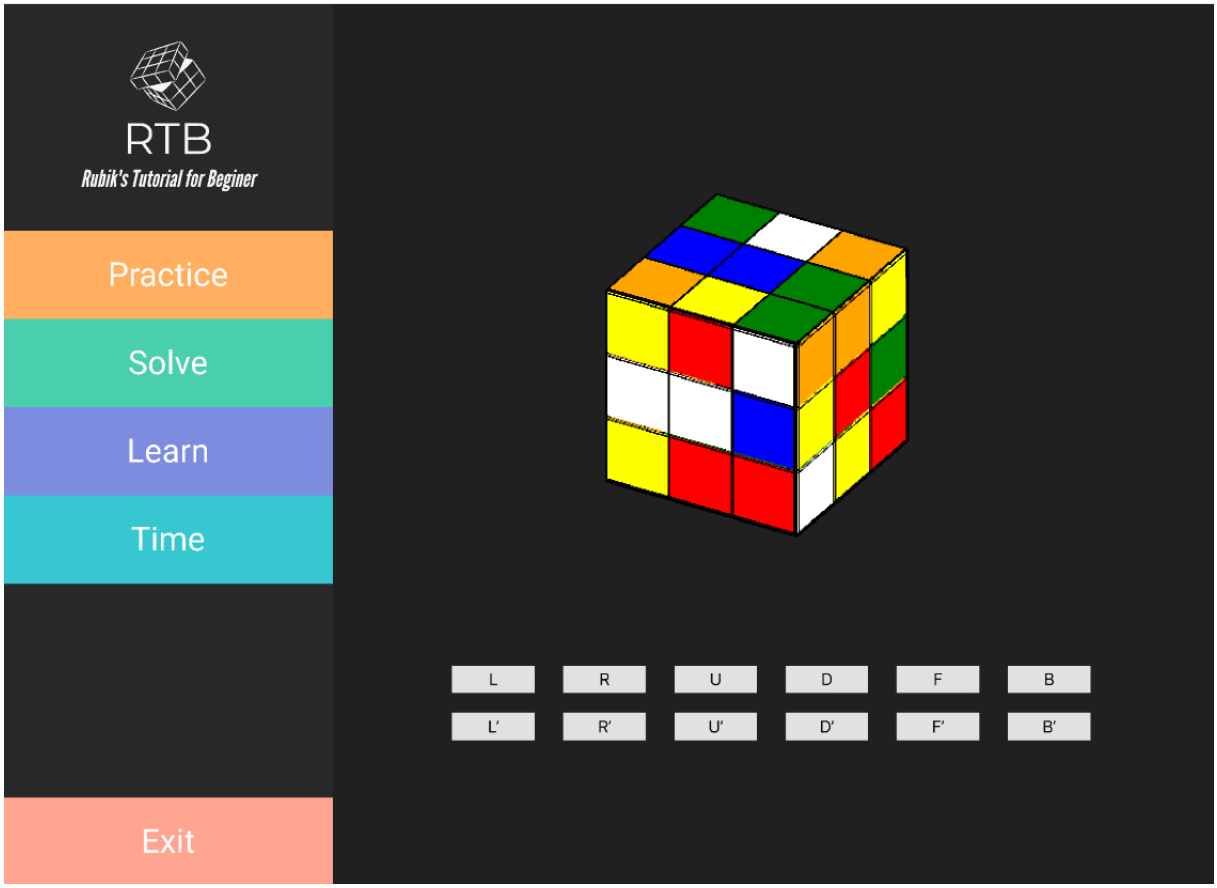
Twisty Polyhedra là trang web mã nguồn mở phục vụ cho việc giải các khối rubik các loại. Điểm mạnh của dự án là công cụ có thể giải được rất nhiều dạng khối rubik. Tuy nhiên đồ họa khối rubik giả lập còn thô sơ, phương pháp nhập dữ liệu khối rubik duy nhất là xoay khối giả lập.

3.2. Thiết kế chương trình

3.2.1. Tab Practice:

Tổng quát: Nhận ra vấn đề rằng không phải người chơi nào cũng có sẵn khối rubik khi tìm hiểu về nó. Nhóm em giả lập để đưa vào ứng dụng một khối rubik ảo có thể xoay và giải như một khối rubik thật.

Giao diện thiết kế ban đầu:



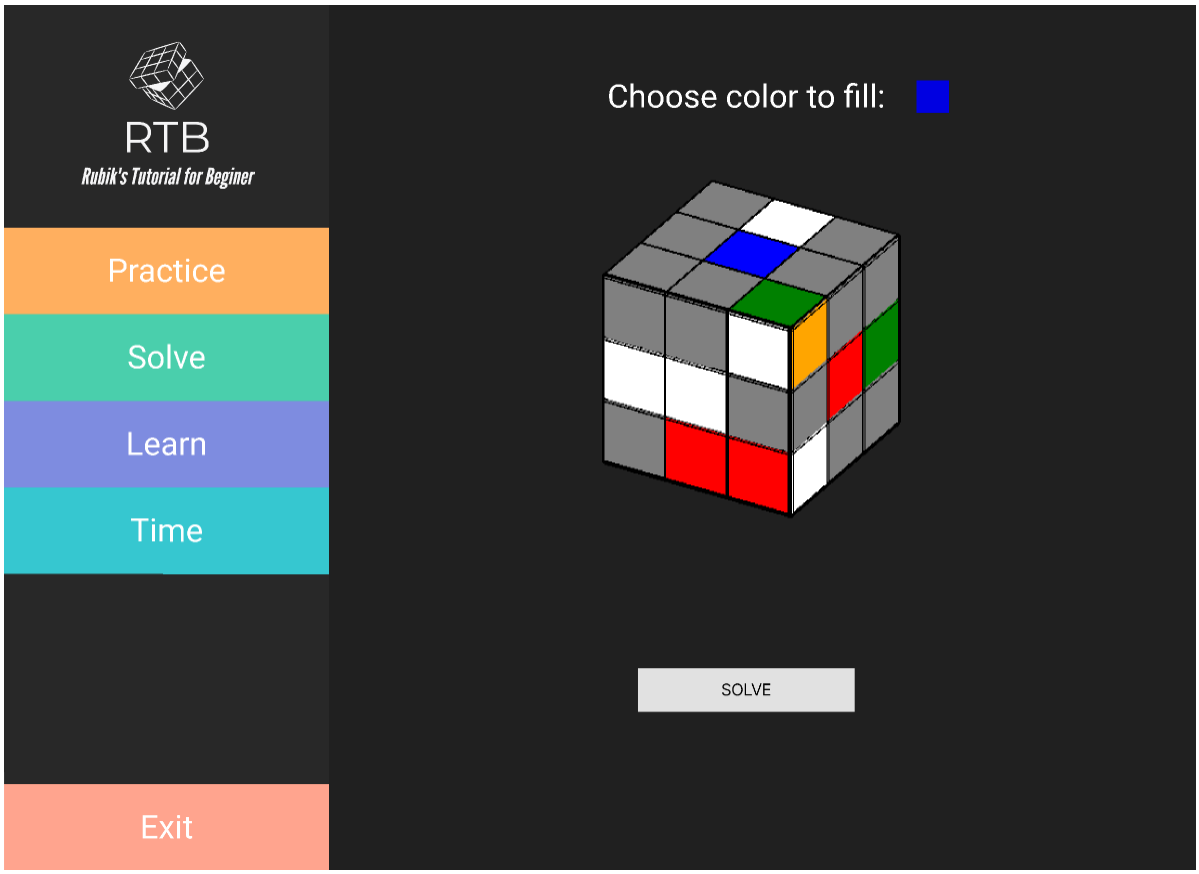
Hình 3-4: Giao diện Practice thiết kế ban đầu

Chức năng: Chương trình đưa ra khối rubik ảo có thể tương tác như một khối rubik thật như xoay từng tầng, quay nguyên khối.

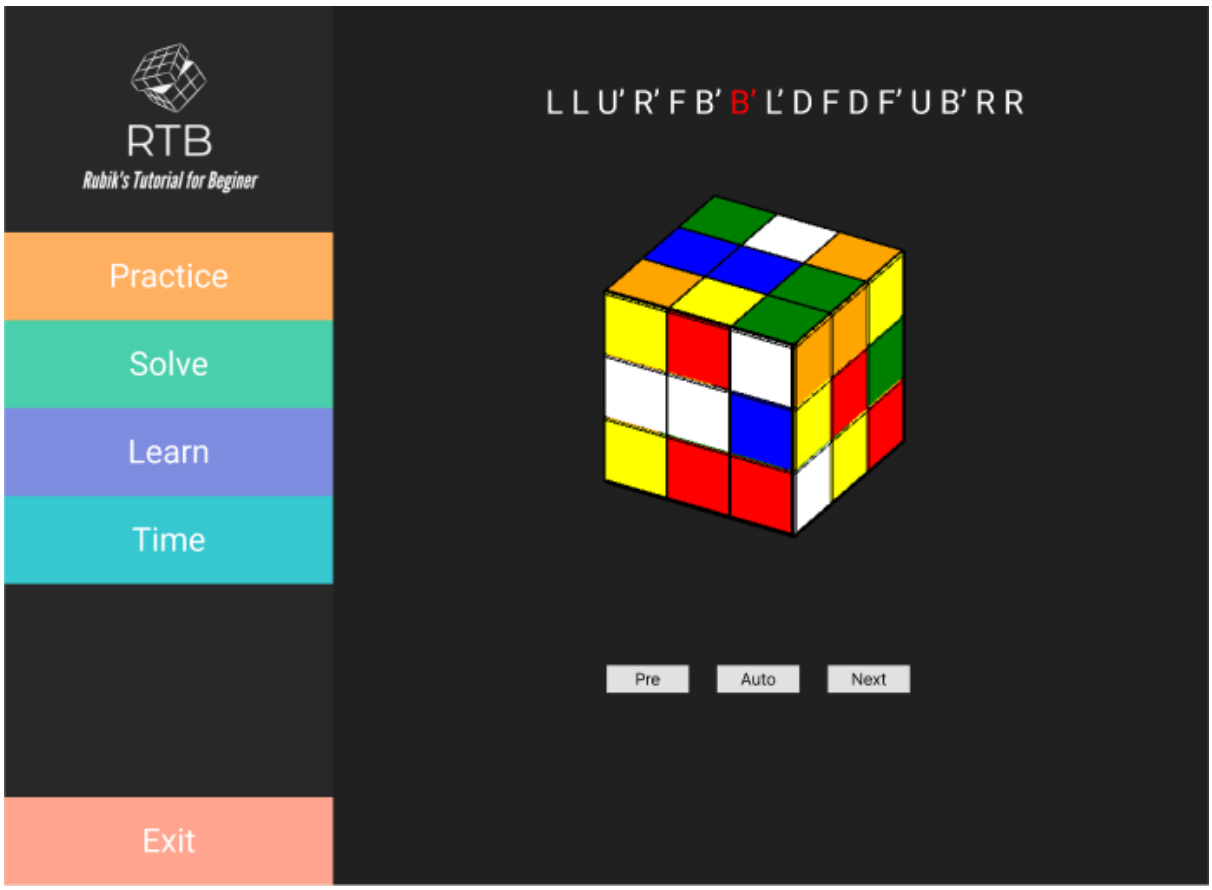
3.2.2. Tab Solve:

Tổng quát: Hiểu được vấn đề đôi lúc người chơi rubik bị bí tại 1 trường hợp cụ thể của khối rubik hay đôi khi chỉ là muốn xem các bước xoay để đưa khối rubik về 6 mặt hoàn chỉnh. Nhóm em đưa ra chức năng để giúp người dùng dễ dàng nhập vào dữ liệu khối rubik và chương trình sẽ đưa ra lời giải cho khối rubik đó đúng theo phương pháp hướng dẫn cho người dùng.

Giao diện thiết kế ban đầu:



Hình 3-5: Giao diện Solve thiết kế ban đầu lúc nhập dữ liệu



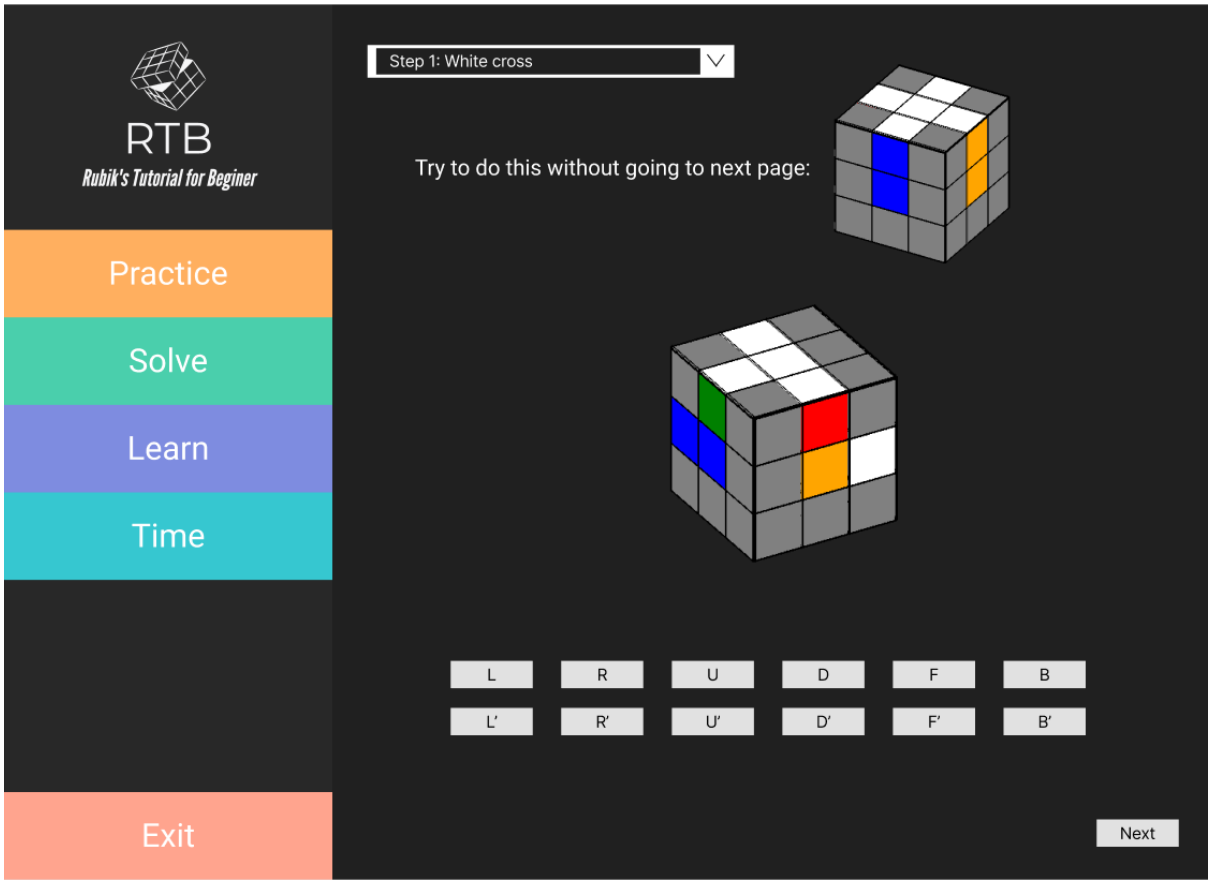
Hình 3-6: Giao diện Solve thiết kế ban đầu lúc giải

Chức năng: Chương trình cho phép người chơi nhập dữ liệu khối rubik mà người chơi muốn giải bằng cách chọn màu và nhấn vào các ô để thay đổi màu tại vị trí ô đó. Sau khi hoàn thành nhập dữ liệu, khi người chơi nhấn vào nút SOLVE, chương trình sẽ hiển thị ra các bước tiếp theo để hoàn thành 6 mặt của khối rubik theo phương pháp CFOP.

3.2.3. Tab Learn:

Tổng quát: Đây là chức năng chính của chương trình, chương trình đưa ra các thuật toán xoay rubik ứng với từng bước để giải quyết khối rubik

Giao diện thiết kế ban đầu:



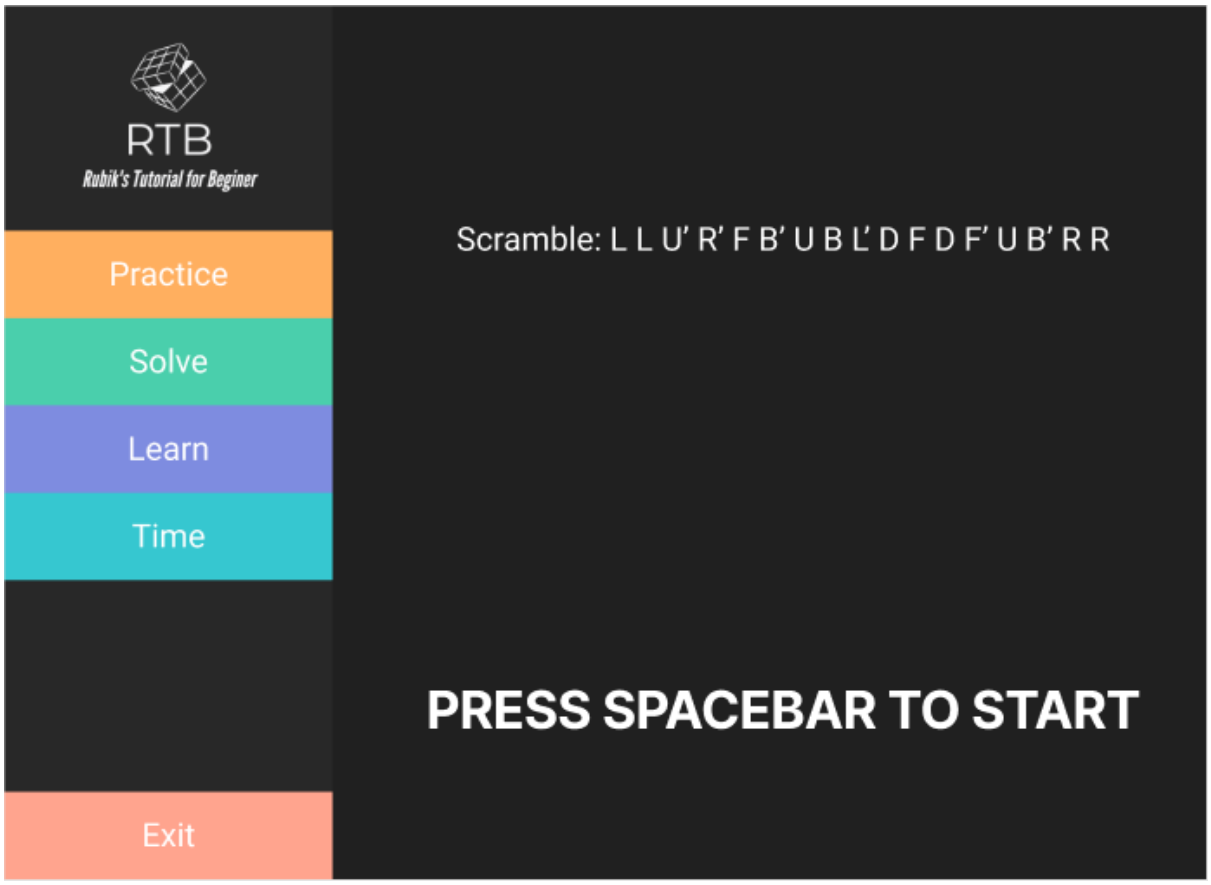
Hình 3-7: Giao diện Learn thiết kế ban đầu

Chức năng: Cho phép người chơi chọn bước giải để đi vào trực tiếp trường hợp mà người chơi chưa nắm rõ thuật toán xoay. Chương trình sẽ đưa ra công thức kèm theo đó là các gợi ý, mẹo nhỏ hay những câu đố (không bắt buộc giải được) để quá trình tập chơi rubik bớt khô khan hơn.

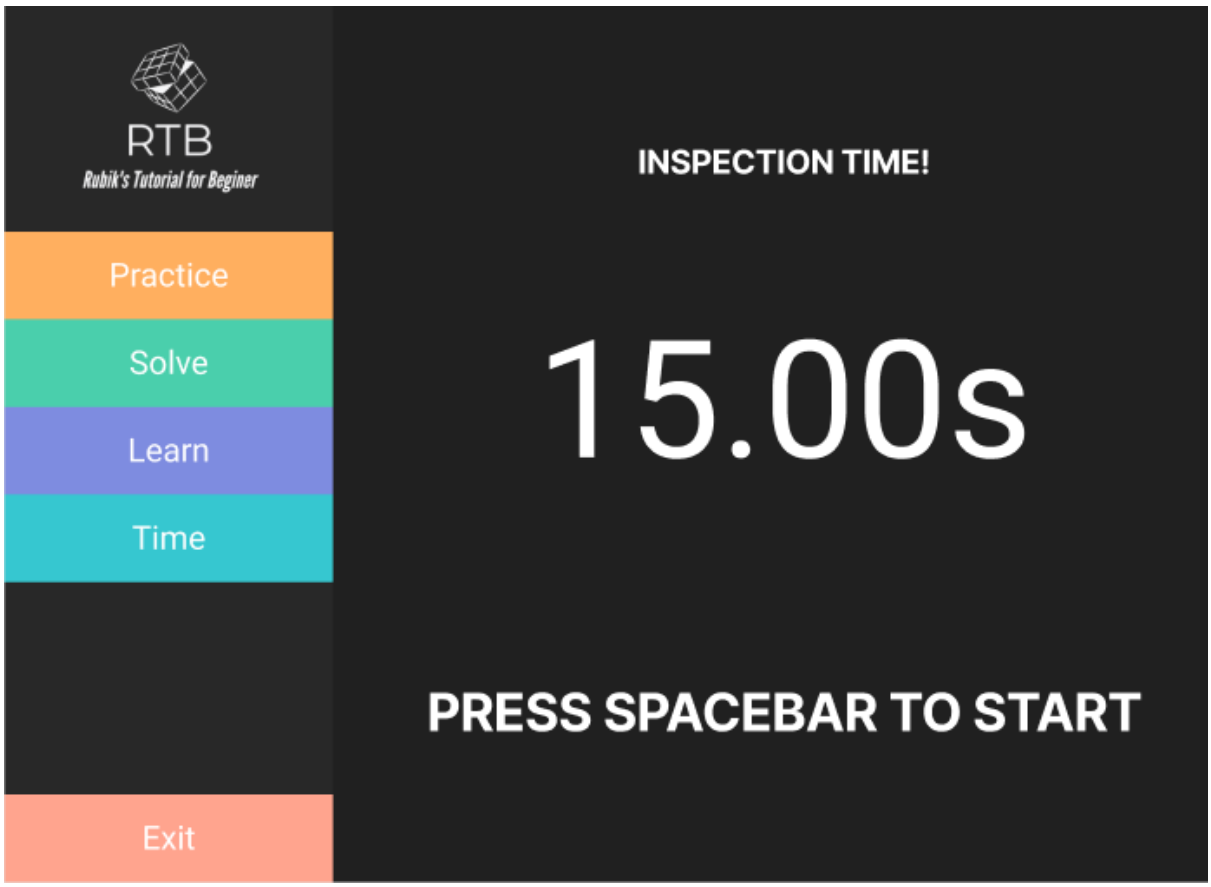
3.2.4. Tab Time:

Tổng quát: Chức năng này sẽ bổ trợ việc quay rubik nhanh hơn sau khi người dùng đã nắm được phương pháp để hoàn thành khối rubik. Người chơi sẽ được đo tốc độ quay rubik (chỉ hỗ trợ rubik thật vì rubik ảo quay khó tương tác nên sẽ tốn nhiều thời gian) và lưu lại kết quả. Từ đó thấy được sự tiến bộ cũng như biết được mức độ thông thạo quay rubik của mình ra sao.

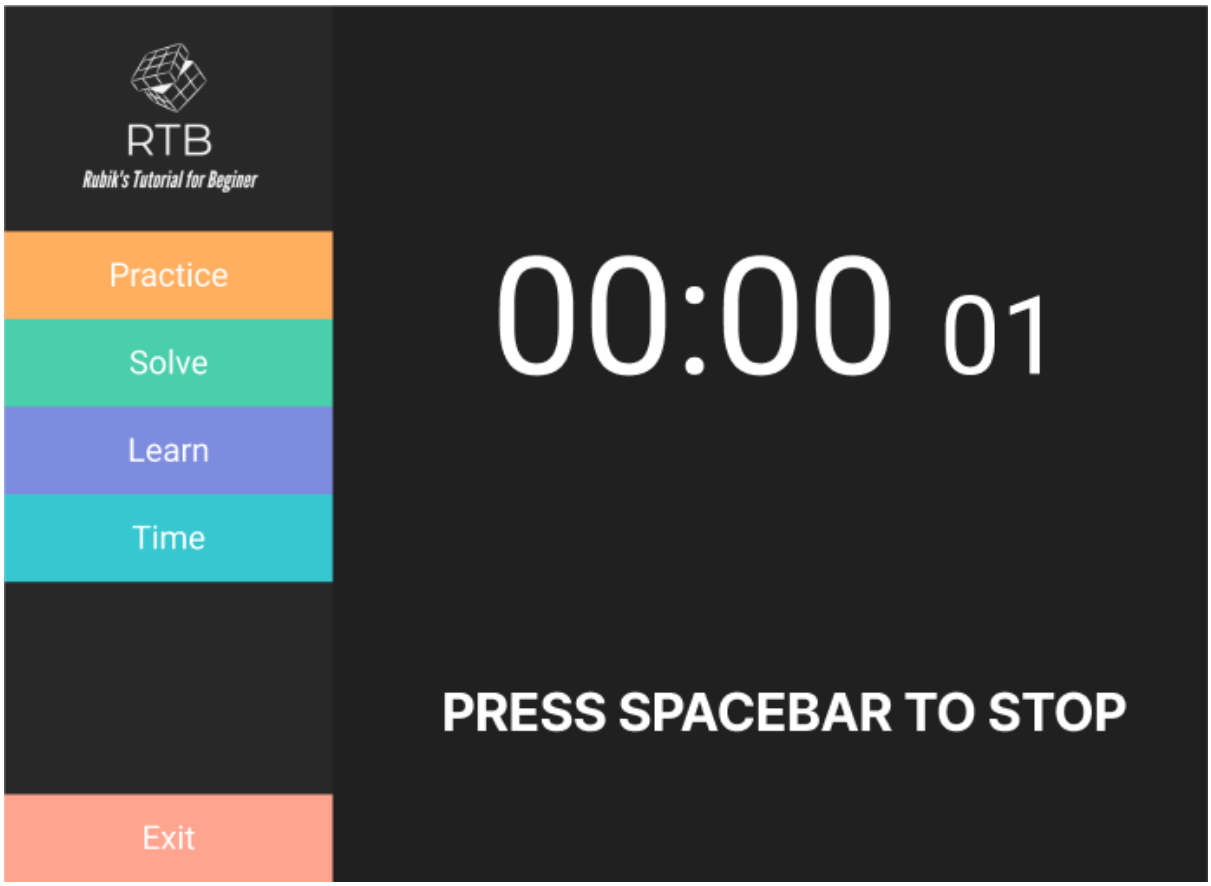
Giao diện thiết kế ban đầu:



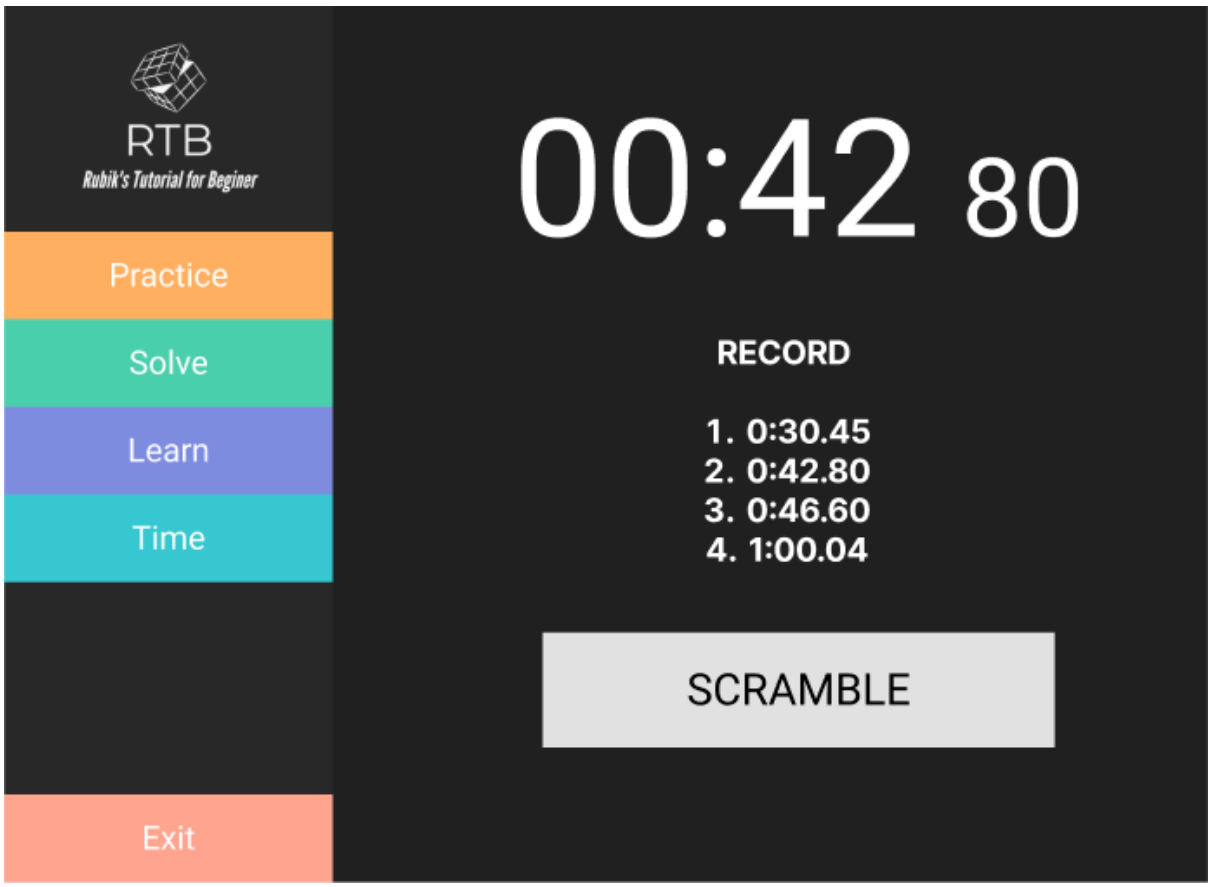
Hình 3-8: Giao diện Time thiết kế ban đầu lúc giải lúc xoay ngẫu nhiên



Hình 3-9: Giao diện Time thiết kế ban đầu lúc giải lúc quan sát khối rubik



Hình 3-10: Giao diện Time thiết kế ban đầu lúc giải lúc bắt đầu tính giờ



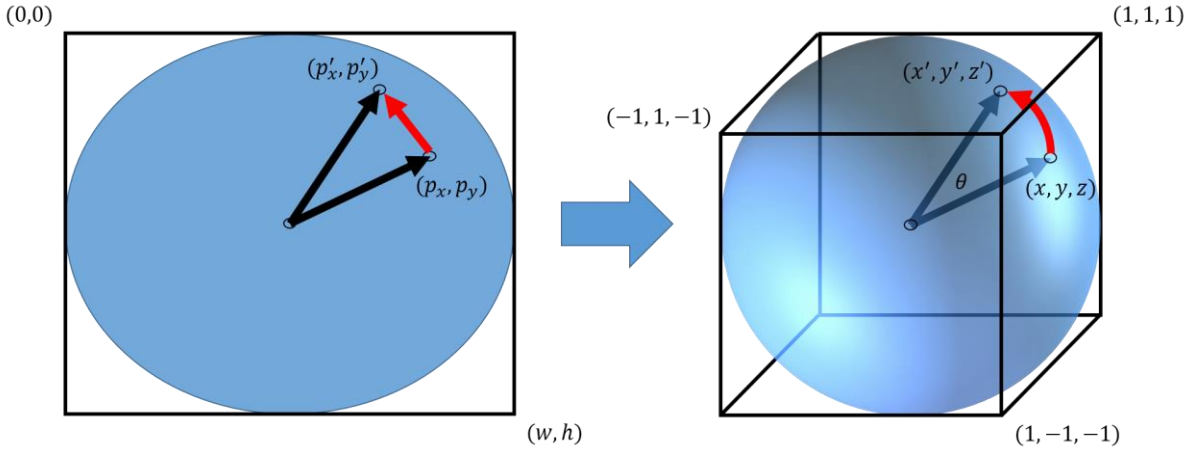
Hình 3-11: Giao diện Time thiết kế ban đầu lúc giải lúc kết thúc

Chức năng: Đưa ra các bước xoay xáo trộn để người chơi xoay theo để tạo ra một trường hợp rubik được coi là ngẫu nhiên. Sau khi người chơi nhấn nút bắt đầu (Space), chương trình cho phép người chơi có 15s để quan sát. Khi người chơi tiếp tục nhấn Space hoặc hết thời gian, chương trình chuyển sang màn hình bắt đầu tính thời gian. Sau khi người chơi nhấn dừng lại, chương trình sẽ xuất ra màn hình thời gian đã ghi lại được đồng thời hiển thị các mốc quay nhanh nhất đã lưu lại được của người chơi (mặc định 5 mốc).

3.3. Một số thuật toán phức tạp

3.3.1. Trackball rotation

Trackball hoạt động bằng cách cố gắng ánh xạ chuyển động nhấp và kéo của chuột trên màn hình tới một chuyển động quay xung quanh bề mặt của hình cầu bán kính 1 được nội tiếp trong hình khối xem của chúng ta trong NDC. Hình 3-9 cho thấy một hình ảnh về điều này:



Hình 3-12: Mô hình trackball rotation 2D và 3D (1)

Một chuyển động nhấp và kéo của chuột trên màn hình của chúng ta (trái) được ánh xạ tới một chuyển động quay xung quanh bề mặt của một hình cầu được nội tiếp khối lập phương xem trong NDC (phải). Các điểm trong vùng màu xanh lam trong sơ đồ bên trái được ánh xạ tới các điểm trên bề mặt hình cầu trong sơ đồ bên phải. Ở đây, (p_x, p_y) và (p'_x, p'_y) được ánh xạ từ tọa độ màn hình thành (x, y) và (x ′, y ′) lần lượt theo NDC; z-NDC được tính bằng một hàm mà chúng ta thảo luận bên dưới. Phương thức Arcball tính toán phép quay sẽ đưa chúng ta từ vectơ vị trí (x, y, z) đến vectơ vị trí (x ′, y ′, z ′) Và sử dụng nó để xoay cảnh.

Chuyển đổi toạ độ pixel sang NDC ([-1,1] coordinate)

Bước 1: (x,y) toạ độ pixel

x giữ nguyên

y = chiều cao cửa sổ - y - 1

Bước 2:

$$x_{ndc} = \frac{(2x - (chiều\ rộng\ cửa\ sổ - 1))}{chiều\ rộng\ cửa\ sổ}$$

$$y_{ndc} = \frac{\left(2x - (chiều\ cao\ cửa\ sổ - 1)\right)}{chiều\ cao\ cửa\ sổ}$$

Giả sử chúng ta nhấp chuột khi nó ở điểm p = (p_x,p_y) trên màn hình theo tọa độ màn hình và di chuyển chuột đến một điểm khác p′= (p′$_x$,p′$_y$) trên màn hình trước khi chúng thả nút chuột. Phương pháp trackball chuyển đổi điểm đầu tiên p và p′ từ tọa độ màn hình sang NDC, bỏ qua thành phần z bị thiếu khỏi tọa độ màn hình hiện tại. Hãy biểu thị tọa độ NDC của p và p′ với p_{ndc} = (x, y) và p′$_{ndc}$ = (x′,y′) tương ứng. Từ đây, Trackball tính toán các tọa độ z cho P_{ndc} và P′$_{ndc}$ bằng cách cố gắng ánh xạ chúng tới các điểm trên bề mặt của hình cầu bán kính 1, có tâm là điểm gốc của hệ thống NDC. Tọa độ z của hai điểm được tính bằng hàm sau:

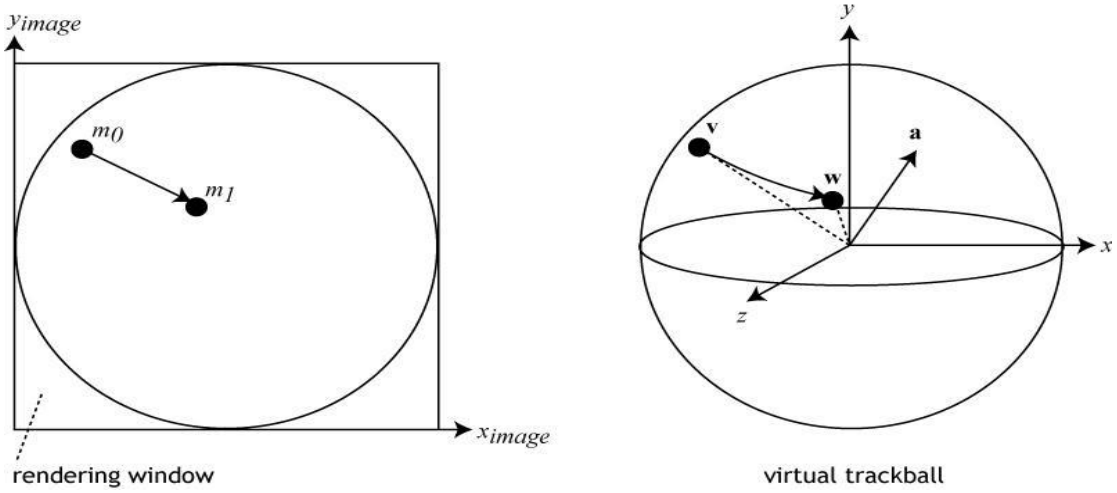
$$z = f(x,y) = \begin{cases} +\sqrt{1 - {x_{ndc}}^2 - {y_{ndc}}^2}, if\ ({x_{ndc}}^2 + {y_{ndc}}^2 \leq 1) \\ 0 \qquad\qquad\qquad\qquad\quad , if\ ({x_{ndc}}^2 + {y_{ndc}}^2 > 1) \end{cases}$$

Trong đồ họa máy tính, một phép quay có thể được định nghĩa theo góc θ và một vectơ đơn vị u để quay. Khá đơn giản khi thấy rằng góc quay θ và vectơ quay u đối với phép quay p_{ndc} thành p'_{ndc} được cho bởi:

$$\theta = cos^{-1}\left(\frac{p_{ndc} \cdot {p'}_{ndc}}{|p_{ndc}\,||{p'}_{ndc}\,|}\right)$$

$$u = p_{ndc} \cdot {p'}_{ndc}$$

θ là góc quay, vector u là trục quay

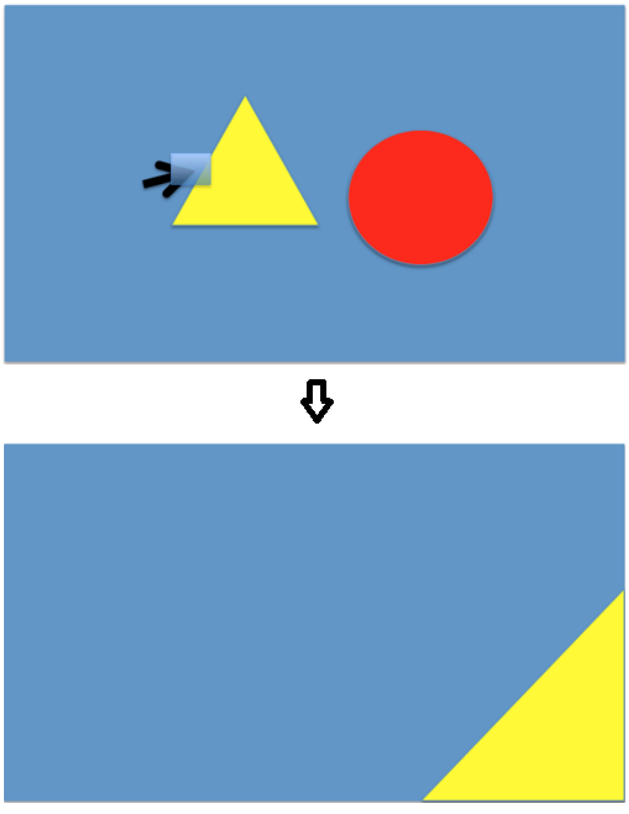


Hình 3-13: Mô hình trackball rotation 2D và 3D (2)

3.3.2. Picking

Một trong những yêu cầu phổ biến nhất trong đồ họa tương tác là người dùng phải nhấp vào đối tượng trên màn hình và trả lại ‘tên’ của đối tượng cho ứng dụng. Điều này được gọi là picking. OpenGL cung cấp cơ chế chọn đối tượng trong cảnh 3D và phần này sẽ chỉ cách phát hiện đối tượng nào nằm dưới con trỏ chuột hoặc trong một vùng hình vuông của cửa sổ OpenGL. Các bước liên quan đến việc phát hiện đối tượng nào ở vị trí nơi con chuột đã được nhấp vào:

1. Lấy tọa độ cửa sổ của con trỏ chuột.
2. Vào chế độ lựa chọn.
3. Xác định lại thể tích xem sao cho chỉ một vùng nhỏ của cửa sổ xung quanh con trỏ được hiển thị.
4. Kết xuất cảnh, bằng cách sử dụng tất cả các bản gốc hoặc chỉ những phần có liên quan đến hoạt động picking.
5. Thoát khỏi chế độ lựa chọn và xác định các đối tượng được hiển thị trên phần nhỏ đó của màn hình.



Hình 3-14: Mô tả Picking tại vị trí con trỏ chuột

Để xác định các đối tượng được kết xuất, phải đặt tên cho tất cả các đối tượng có liên quan trong cảnh. API OpenGL cho phép đặt tên cho các nguyên thủy hoặc tập hợp các nguyên thủy (đối tượng). Khi ở chế độ lựa chọn, một chế độ kết xuất đặc biệt, không có đối tượng nào thực sự được hiển thị trong khung đệm. Thay vào đó, tên của các đối tượng (cộng với thông tin về độ sâu) được thu thập trong một mảng. Đối với các đối tượng không được đặt tên, chỉ thông tin về độ sâu được thu thập.

Sử dụng thuật ngữ OpenGL, một 'lần truy cập' xảy ra bất cứ khi nào một bản gốc được hiển thị trong chế độ lựa chọn. Bản ghi lượt truy cập được lưu trữ trong bộ đệm lựa chọn. Khi thoát khỏi chế độ lựa chọn OpenGL trả về bộ đệm lựa chọn với tập hợp các bản ghi lần truy cập. Kể từ khi OpenGL cung cấp thông tin độ sâu cho mỗi lần truy cập, ứng dụng sau đó có thể dễ dàng phát hiện đối tượng nào ở gần hơn với người dùng.

Hit Record Contents	Description
0	No names have been stored for the first hit
4.2822e+009	Minimum depth for first hit
4.28436e+009	Maximum depth for first hit
0	Number of names for the second hit
4.2732e+009	Minimum depth for second hit
4.27334e+009	Maximum depth for second hit
6	A single name of the second hit
2	Number of names for the third hit
4.27138e+009	Minimum depth for third hit
4.27155e+009	Maximum depth for third hit
2	First name of the third hit
5	Second name of the third hit

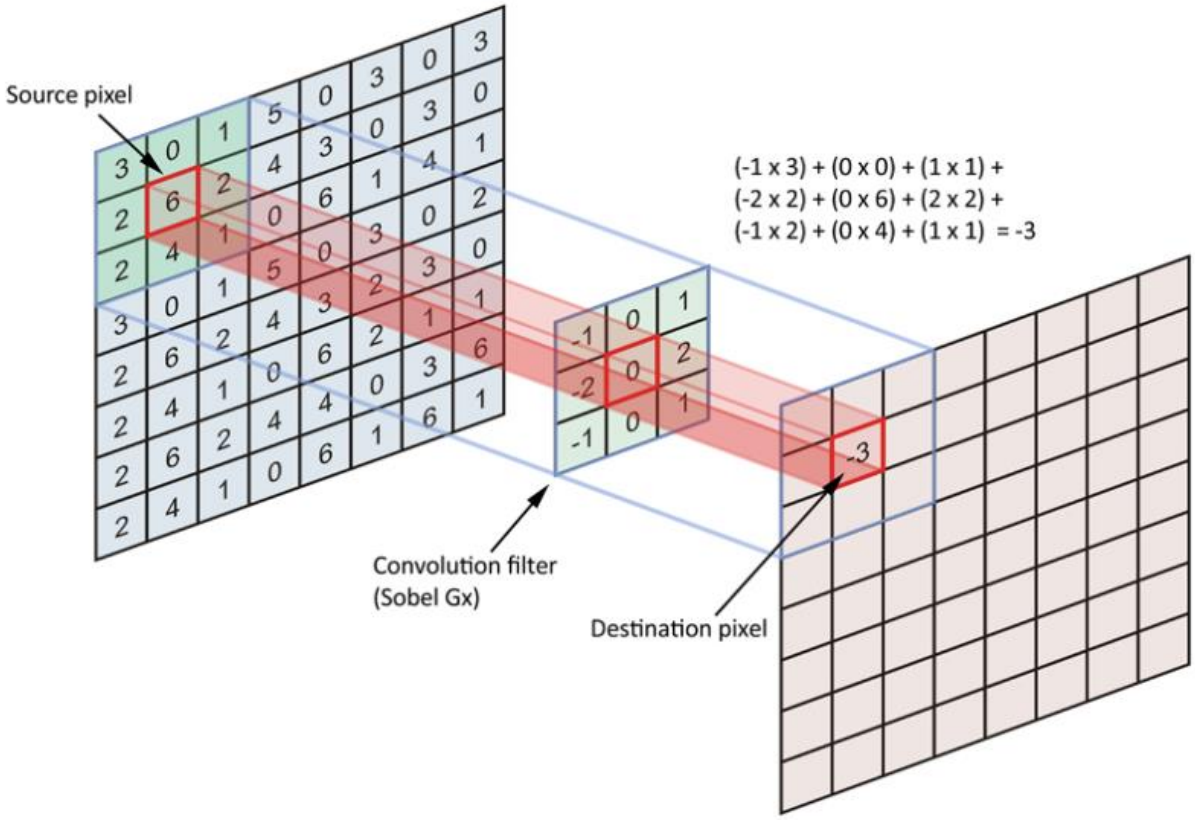
Bảng 3-1: Selection Buffer

3.3.3. Xử lí ảnh

Convolution: là các phép tính toán giữa ảnh và toán tử (kernel).

Kernel: là mảng 2 chiều (ma trận) có kích thước cố định do người dùng chỉ định.

Anchor point: là điểm neo nằm bên trong kernel, thường là vị trí trung tâm kernel.



Hình 3-15: Xử lí ảnh qua filter kernel 3x3

MedianBlur:

Lọc Median hay còn gọi là lọc trung vị khác với bộ lọc trung bình ở chỗ nó không xảy ra bất cứ phép tính toán gì, nó chỉ đơn giản là chỉ ra đâu là giá trị nằm ở vị trí trung bình trong các giá trị đầu vào được thiết lập bởi khung lọc rồi lấy giá trị đó thế vào tâm của khung lọc trong ảnh gốc để tạo ra giá trị ảnh mới. Hiệu quả của phép lọc có khí không rõ ràng, khó có thể nhận ra được sự thật đằng sau phép biến đổi vì nó chỉ thay đổi một số điểm ảnh thôi chứ không phải toàn bộ các điểm ảnh nhưng nó có ưu thế là sự sửa chữa mạnh mẽ một số điểm bất thường trên ảnh mà các bộ lọc khác không làm được hoặc dễ dàng bỏ qua.

Thresholding

Nếu pixel có giá trị lớn hơn giá trị ngưỡng thì nó được gán 1 giá trị (thường là 1), ngược lại nhỏ hơn giá trị ngưỡng thì nó được gán 1 giá trị khác (thường là 0).

Trong opencv, ngưỡng là một số nằm trong đoạn từ 0 đến 255. Giá trị ngưỡng sẽ chia tách giá trị độ xám của ảnh thành 2 miền riêng biệt. Miền thứ nhất là tập hợp các điểm ảnh có giá trị nhỏ hơn giá trị ngưỡng. Miền thứ hai là tập hợp các các điểm ảnh có giá trị lớn hơn hoặc bằng giá trị ngưỡng.

Đầu vào của một thuật toán phân ngưỡng trong opencv thường có input là ảnh nguồn (source image) và giá trị ngưỡng. Đầu ra là ảnh đích đã được phân ngưỡng (destination image).

Simple thresholding

Simple Thresholding thực hiện phân ngưỡng bằng cách thay thế giá trị lớn hơn hoặc bằng và giá trị bé hơn giá trị ngưỡng bằng một giá trị mới.

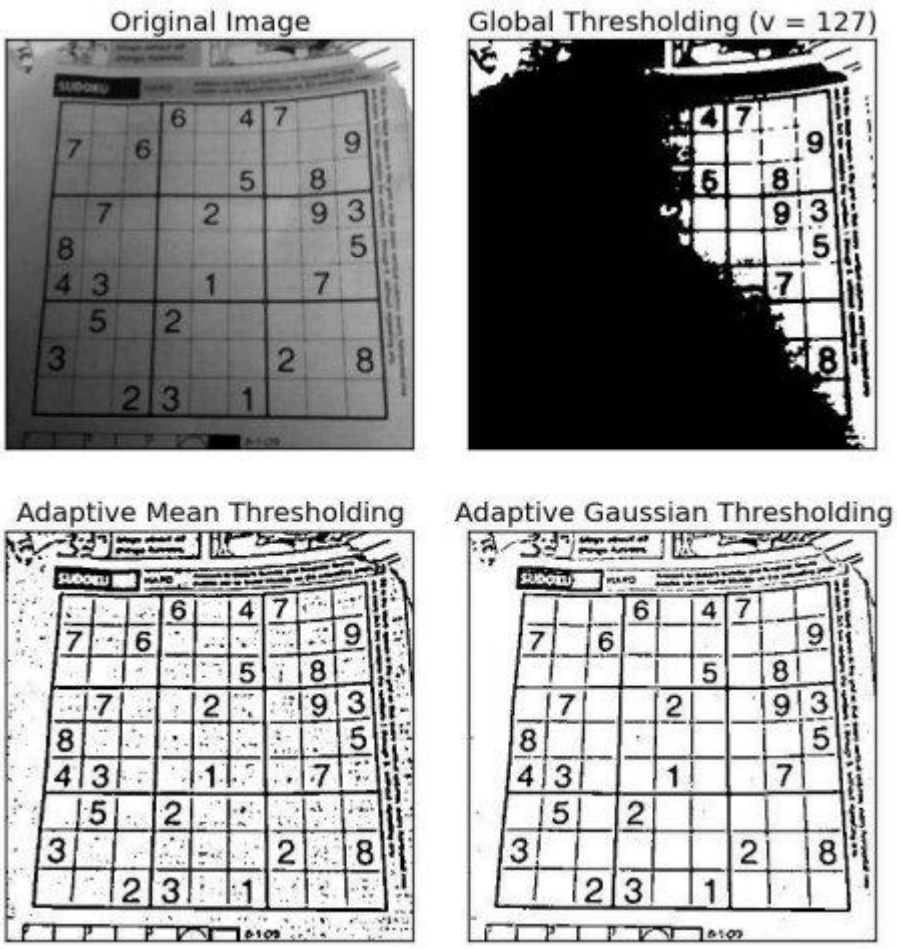
Adaptive thresholding

Thuật toán simple thresholding hoạt động khá tốt. Tuy nhiên, nó có 1 nhược điểm là giá trị ngưỡng bị/được gán toàn cục. Thực tế khi chụp, hình ảnh chúng ta nhận được thường bị ảnh hưởng của nhiễu, ví dụ như là bị phơi sáng, bị đèn flash, …

Một trong những cách được sử dụng để giải quyết vấn đề trên là chia nhỏ bức ảnh thành những vùng nhỏ (region), và đặt giá trị ngưỡng trên những vùng nhỏ đó -> adaptive thresholding ra đời. Opencv cung cấp cho chúng ta hai cách xác định những vùng nhỏ

ADAPTIVE_THRESH_MEAN_C: Tính trung bình các láng giềng xung quanh điểm cần xét trong vùng blockSize * blockSize trừ đi giá trị hằng số C.

ADAPTIVE_THRESH_GAUSSIAN_C: Nhân giá trị xung quanh điểm cần xét với trọng số gauss rồi tính trung bình của nó, sau đó trừ đi giá trị hằng số C.



Hình 3-16: So sánh hiệu quả threshold và adaptivethreshold

Canny detection

Trong hình ảnh, thường tồn tại các thành phần như: vùng trơn, góc / cạnh và nhiễu. Cạnh trong ảnh mang đặc trưng quan trọng, thường là thuộc đối tượng trong ảnh (object). Do đó, để phát hiện cạnh trong ảnh, giải thuật Canny là một trong những giải thuật phổ biến / nổi tiếng nhất trong Xử lý ảnh

Giải thuật phát hiện cạnh Canny gồm 4 bước chính sau:

• Giảm nhiễu: Làm mờ ảnh, giảm nhiễu dùng bộ lọc Gaussian kích thước 5x5. Kích thước 5x5 thường hoạt động tốt cho giải thuật Canny. Tuy nhiên vẫn có thể thay đổi kích thước khác phù hợp hơn.

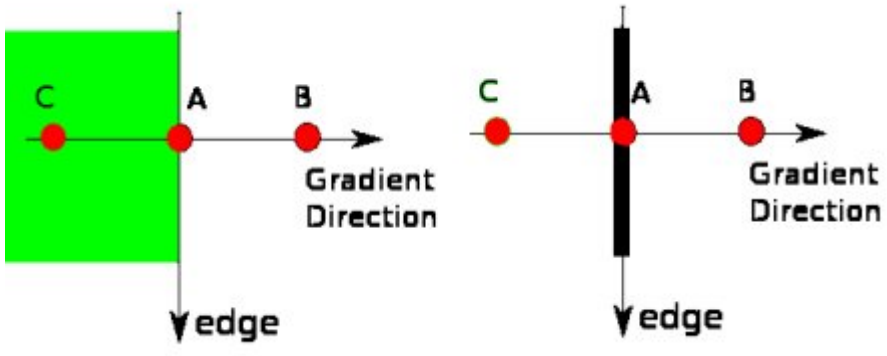
• Tính Gradient và hướng gradient: dùng bộ lọc Sobel X và Sobel Y (3x3) để tính được ảnh đạo hàm Gx và Gy. Sau đó, tiếp tục tính ảnh Gradient và góc của Gradient theo công thức. Ảnh đạo hàm Gx và Gy là ma trận (ví dụ: 640x640), thì kết quả tính ảnh đạo hàm Edge Gradient cũng là một ma trận (640x640), mỗi pixel trên ma trận này thể

hiện độ lớn của biến đổi mức sáng ở vị trí tương ứng trên ảnh gốc. Tương tự, ma trận Angle cũng có cùng kích thước (640x640), mỗi pixel trên Angle thể hiện góc, hay nói cách khác là hướng của cạnh. Nếu góc gradient là 0 độ, thì cạnh trên ảnh sẽ là một đường thẳng đứng (tức tạo góc 90 độ so với trục hoành) (vuông góc hướng gradient). Khi tính toán, giá trị hướng gradient sẽ nằm trong đoạn [-180, 180] độ, không giữ nguyên các góc này mà gom các giá trị này về 4 bin đại diện cho 4 hướng: hướng ngang (0 độ), hướng chéo bên phải (45 độ), hướng dọc (90 độ) và hướng chéo trái (135 độ). Hướng vector gradient luôn vuông góc với các cạnh. Nó được làm tròn thành một trong bốn góc đại diện cho các hướng dọc, ngang và hai đường chéo.

$$Edge\_Gradient\ (G) = \sqrt{G_x^2 + G_y^2}$$

$$Angle(\theta) = \tan^{-1}\left(\frac{G_y}{G_x}\right)$$

• Non-maximum Suppression (viết tắt NMS): loại bỏ các pixel ở vị trí không phải cực đại toàn cục. Ở bước này, dùng một filter 3x3 lần lượt chạy qua các pixel trên ảnh gradient. Trong quá trình lọc, xem xét xem độ lớn gradient của pixel trung tâm có phải là cực đại (lớn nhất trong cục bộ - local maximum) so với các gradient ở các pixel xung quanh. Nếu là cực đại, ghi nhận sẽ giữ pixel đó lại. Còn nếu pixel tại đó không phải là cực đại lân cận, sẽ thiết lập độ lớn gradient của nó về 0. Chỉ so sánh pixel trung tâm với 2 pixel lân cận theo hướng gradient:



Hình 3-17: Mô phỏng thuật toán phát hiện cạnh

Điểm A nằm trên cạnh (theo phương thẳng đứng). Hướng vector Gradient vuông góc đối với cạnh. Điểm B và C theo hướng vector gradient. Vì vậy, điểm A được kiểm tra với điểm B và C để xem nó có tạo thành cực đại cục bộ hay không. Nếu vậy, nó được

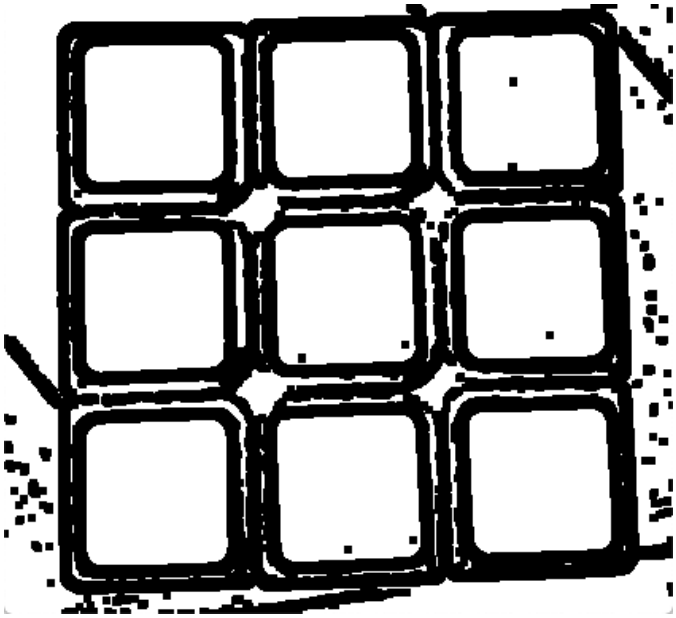
xem xét cho giai đoạn tiếp theo, nếu không, nó sẽ bị loại bỏ (đưa về 0). Tóm lại, kết quả nhận được là một hình ảnh nhị phân với "các cạnh mỏng".

• Lọc ngưỡng: xét các pixel dương trên mặt nạ nhị phân kết quả của bước trước. Nếu giá trị gradient vượt ngưỡng max_val thì pixel đó chắc chắn là cạnh. Các pixel có độ lớn gradient nhỏ hơn ngưỡng min_val sẽ bị loại bỏ. Còn các pixel nằm trong khoảng 2 ngưỡng trên sẽ được xem xét rằng nó có nằm liền kề với những pixel được cho là "chắc chắn là cạnh" hay không. Nếu liền kề thì giữ, còn không liền kề bất cứ pixel cạnh nào thì loại. Sau bước này có thể áp dụng thêm bước hậu xử lý loại bỏ nhiễu (tức những pixel cạnh rời rạc hay cạnh ngắn) nếu muốn.

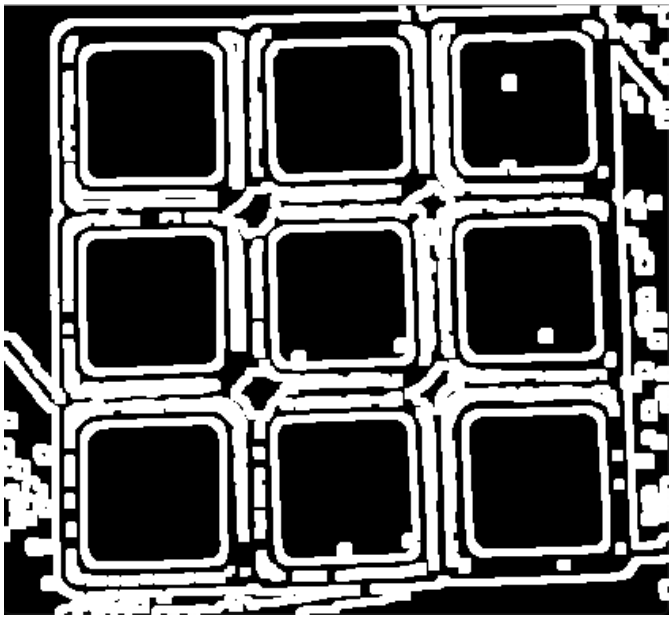
Ảnh minh họa về ngưỡng lọc áp dụng nhận diện các ô màu khối rubik 3x3:

Hình 3-18: Ảnh nguồn chụp khối rubik 3x3

Hình 3-19: Ảnh xám sau khi qua bộ lọc medianBlur (kernel size 5x5)



Hình 3-20: Phân ngưỡng động (Adaptivethreshold)



Hình 3-21: Phát hiện cạnh (Canny Detection)

Dùng hàm findcontours(opencv) tìm tất cả đường viền của 1 ảnh nhị phân.

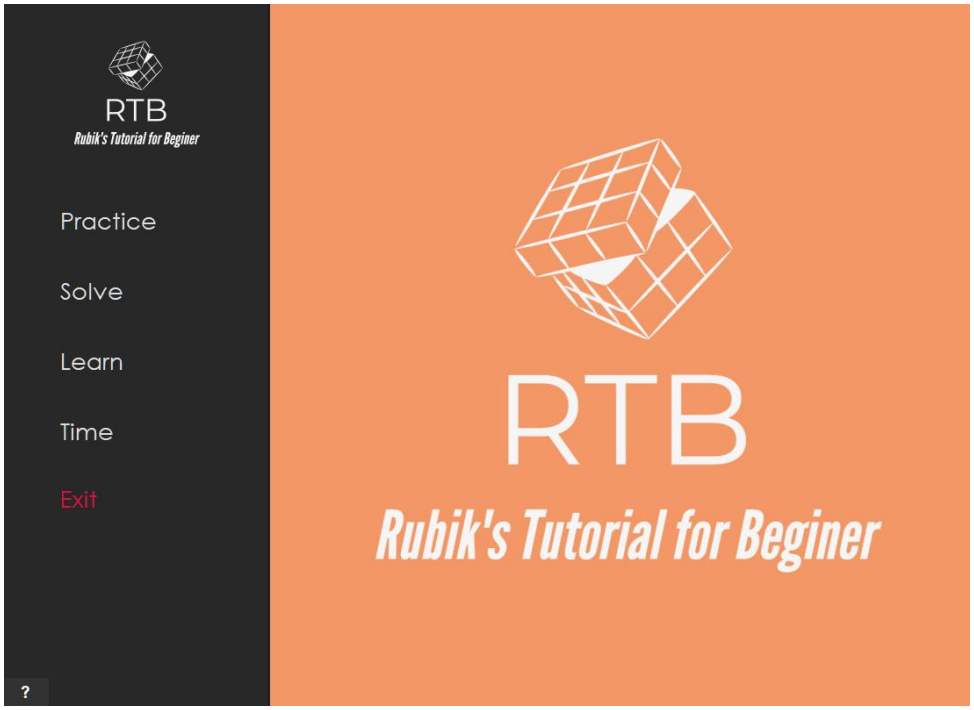
Contour là "tập các điểm-liên-tục tạo thành một đường cong (curve) (boundary), và không có khoảng hở trong đường cong đó, đặc điểm chung trong một contour là các các điểm có cùng /gần xấp xỉ một giá trị màu, hoặc cùng mật độ. Contour là một công cụ hữu ích được dùng để phân tích hình dạng đối tượng, phát hiện đối tượng và nhận dạng đối tượng". Xấp xỉ các đường viền vừa tìm được lọc ra những xấp xỉ hình vuông.

Hình 3-22: Kết quả nhận diện các ô của khối rubik

4. KẾT QUẢ HIỆN THỰC VÀ ĐÁNH GIÁ

4.1. Giao diện chương trình và chức năng

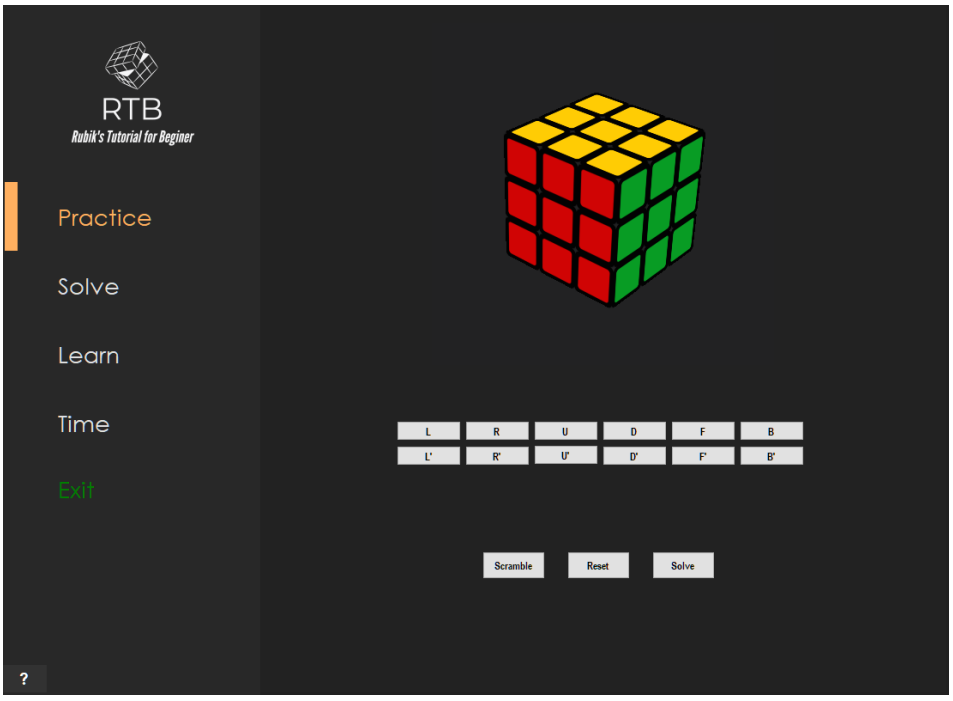
Giao diện Home khi khởi động chương trình:



Hình 4-1: Giao diện Home

Đây là giao diện Home. Tại đây người dùng có thể truy cập vào các tab như Practice, Solve, Learn, Time và thoát khỏi chương trình thông qua nút Exit.

Giao diện tab Practice:



Hình 4-2: Giao diện Practice

Tab Practice có các chức năng chính sau:

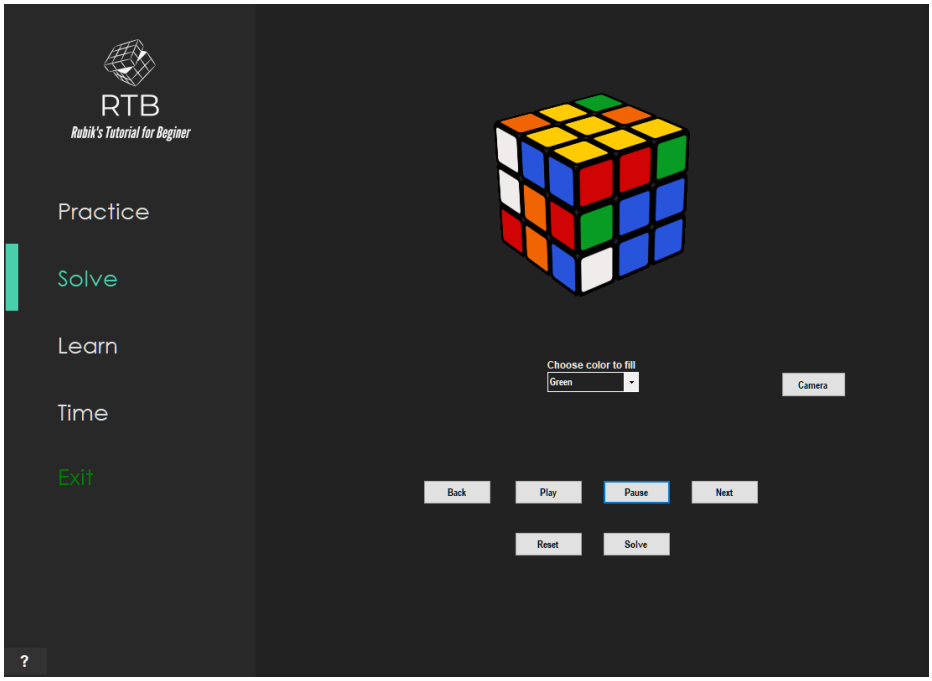
Dựng khối rubik ảo để người dùng có thể quan sát và tương tác. Người dùng có thể tương tác qua các nút xoay dưới khối rubik ;qua chuột bằng cách kéo chuột vùng xung quay khối rubik để xoay nguyên khối hay kéo chuột tại các vị trí trên khối rubik để xoay các tầng.

Solve: giải khối rubik mà người chơi đã xáo trộn theo phương pháp 7 bước nhưng với tốc độ xoay khá nhanh.

Scramble: làm khối rubik ảo tự xáo trộn.

Reset: làm khối rubik về trạng thái hoàn thành như ban đầu.

Giao diện tab Solve:

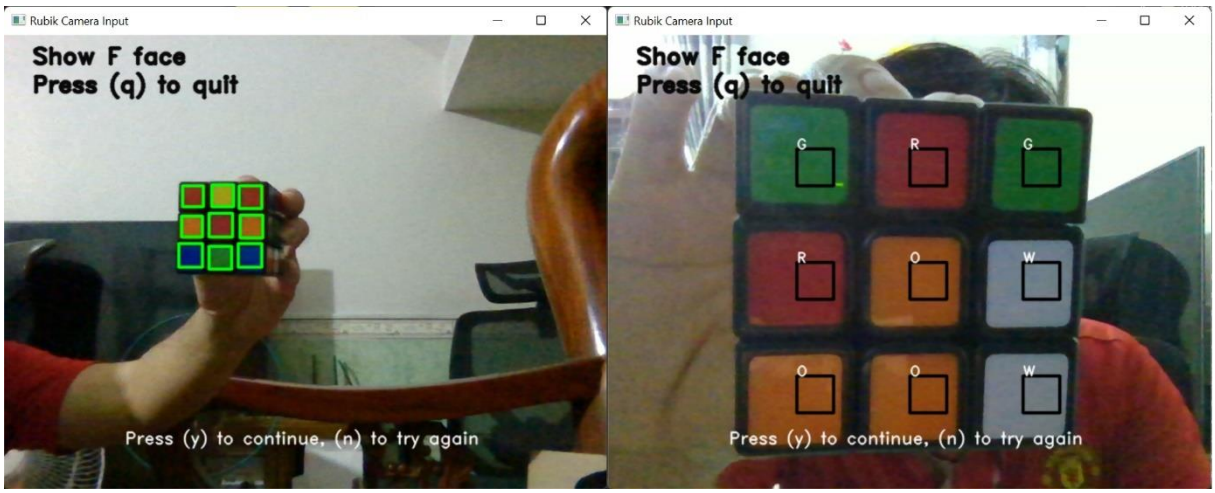


Hình 4-3: Giao diện Solve

Tab Solve có các chức năng chính sau:

Choose color to fill: Chọn 1 trong 6 màu để chỉnh sửa màu của từng viên trong khối rubik ảo.

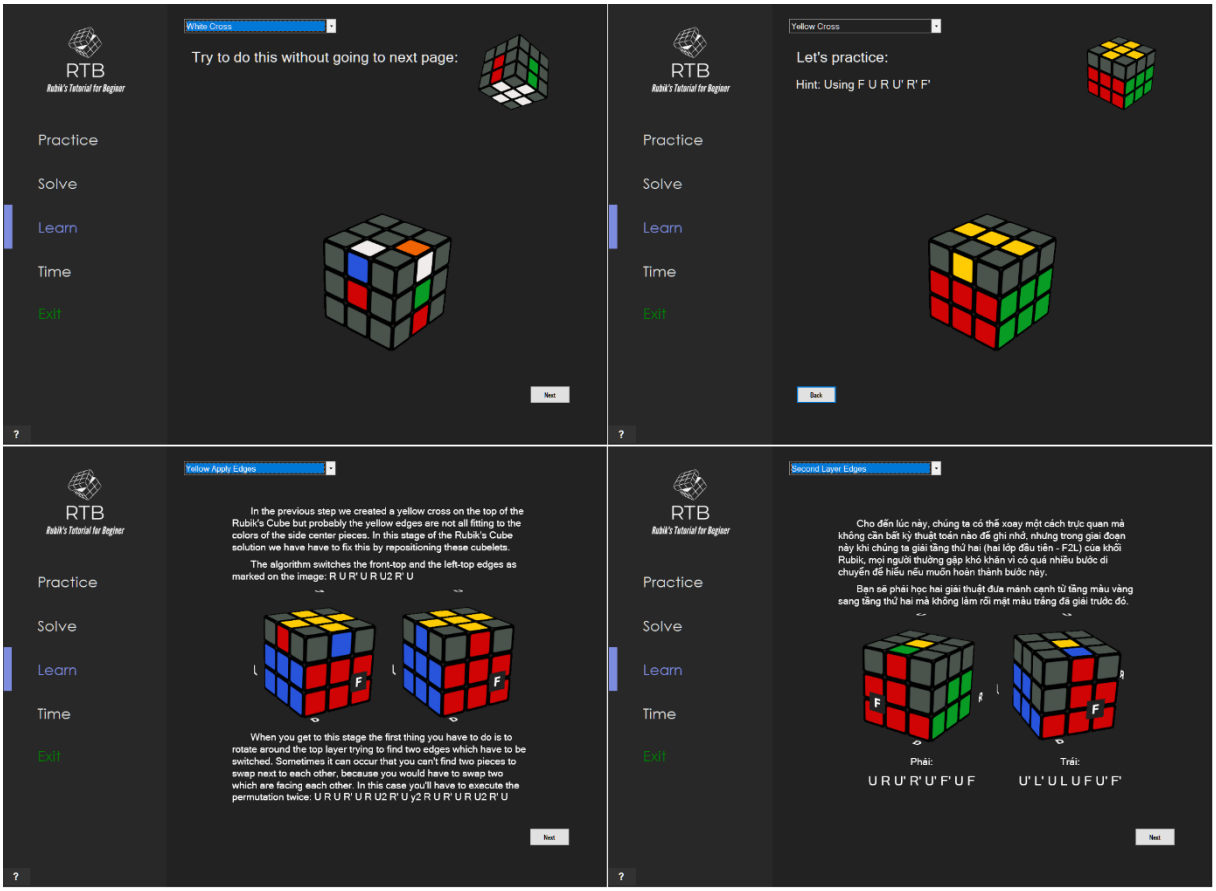
Camera: sử dụng camera để nhận diện khối rubik thật của người chơi, sau đó truyền thông tin để chỉnh sửa khối rubik ảo của chương trình.



Hình 4-4: Giao diện camera lấy dữ liệu từ rubik thực

Solve: giải khối rubik ảo phía trên. Người dùng sử dụng nút Play để chương trình tự giải các bước đến khi hoàn thành, Pause để ngưng tại bước đang thực hiện, Next để chạy 1 bước tiếp theo, Back để quay về 1 bước trước đó.

Giao diện tab Learn:



Hình 4-4: Giao diện Learn

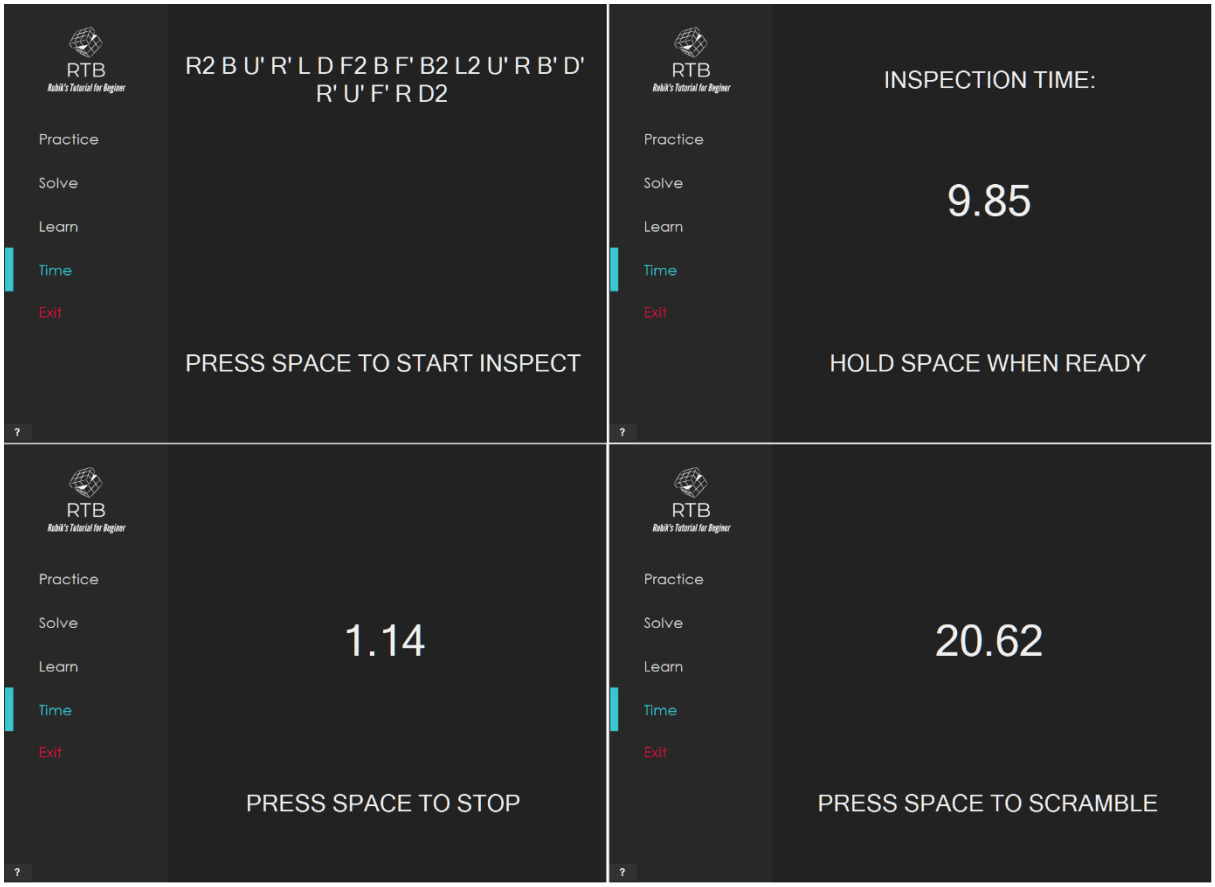
Tab Learn có các chức năng chính sau:

ComboBox: Lựa chọn level mà người dùng muốn học, chương trình sẽ hướng dẫn cách xoay rubik tại level đó.

Khối rubik ảo để người chơi rèn luyện xoay theo khối rubik mẫu.

Các bài hướng dẫn có cả tiếng Anh và tiếng Việt cho người dùng dễ dàng tiếp cận phương pháp giải khối rubik mà chương trình muốn truyền tải.

Giao diện tab Time:



Hình 4-5: Giao diện Time

Tab Time có các chức năng chính sau:

Scramble: Hiển thị các bước để người dùng xáo trộn khối rubik thực.

Inspection time: Hiển thị thời gian quan sát rubik còn lại cho người dùng.

Timing: Đo thời gian người chơi giải xong khối rubik.

4.2. Đánh giá

So với mục tiêu ban đầu đề ra là viết chương trình Hướng dẫn chơi rubik bằng thư viện đồ họa openGL thì chương trình Rubik's Totorial for Beginer đã đáp ứng được kỳ vọng đó. Một cách tổng quát, chương trình đã có thể dựng đồ họa khối rubik ảo, đồng thời hỗ trợ người chơi trong việc tập chơi rubik thông qua hướng dẫn. Việc này có ý nghĩa nhiều trong giáo dục. Đã có các chương trình hướng dẫn chơi rubik khác nhưng không có nhiều chức năng. Đối với RTB, chương trình hỗ trợ tận tình cho người dùng như tương tác với khối rubik ảo hay luyện tập vận dụng các thuật toán quay bước từng bước. Đây là điểm có phần nổi bật hơn.

Không dừng lại ở đó, chương trình còn sử dụng thư viện xử lí thị giác máy tính để người dùng có thể tương tác tốt hơn với chương trình qua cách nhập dữ liệu từ khối rubik thực nhanh và chính xác.

5. TỔNG KẾT

5.1. Kết luận

Đề tài đã xây dựng được chương trình hướng dẫn chơi rubik trong đó có mô phỏng khối rubik ảo để người dùng tương tác và luyện tập. Đóng góp của đề tài có ý nghĩa trong giáo dục và trong giải trí. Dựa vào đó, người dùng có thêm những kiến thức, trãi nghiệm giúp việc chơi rubik trở nên dễ dàng và giải trí hơn.

5.1.1. Ưu điểm

Những ưu điểm của chương trình:

- Chương trình dựng đồ họa khối rubik ảo 3 chiều sinh động và đẹp mắt
- Người dùng có thể tương tác dễ dàng với khối rubik ảo
- Chỉ ra cách giải khối rubik từng bước một
- Tương tác tốt với cả rubik thực thông qua camera máy tính
- Hướng dẫn cả lí thuyết và thực hành cho người dùng
- Có chức năng tính toán thời gian giải rubik cho người dùng theo dõi được mức độ phát triển của bản thân

5.1.2. Khuyết điểm

Tuy vậy, chương trình khó tránh khỏi những thiếu sót:

- Giao diện có phần lỗi thời, không được bắt mắt
- Mặc dù có thể lấy dữ liệu từ rubik thực nhưng không lấy được nhiều mặt cùng một lúc
- Chương trình chưa có chức năng lưu lại thời gian giải rubik cho người chơi

5.2. Hướng phát triển

Mặc dù đã đạt được những kết quả nhất định, tuy nhiên, còn rất nhiều điều cần làm để cải thiện chương trình tốt hơn, đạt đến mức có thể sử dụng rộng rãi. Với hướng tiếp cận với thị giác máy tính như chức năng lấy thông tin rubik thực của chương trình, hệ thống hoàn toàn có thể được bổ sung thêm nhiều chức năng. Trong tương lai, chương trình có thể nâng cấp đồ họa giao diện bằng cách sử dụng thêm các thư viện đồ họa có mã nguồn mở dành cho giao diện winform. Một số tính năng cần được bổ sung như hỗ trợ trên website hay mobile, thay đổi giao diện sáng tối, cho người dùng tùy chọn thông số của rubik hay phát triển cho những loại rubik khác như 4x4, 5x5…

TÀI LIỆU THAM KHẢO

[1] Edward Angel andDave Shreiner, "Interactive computer graphics a top-down approach using OpenGL", 2011

[2] https://rubikscu.be (Truy cập lần cuối ngày 15/5/2022)

[3] "OpenTK", https://opentk.net/index.html (Truy cập lần cuối ngày 15/5/2022)

[4] "Rubik's Cube and Twisty Puzzle Wiki", https://ruwix.com (Truy cập lần cuối ngày 15/5/2022)

[5] Nguyen Cong ,"TỔNG QUAN VỀ KHỐI RUBIK 3X3 VÀ CÁC KÝ HIỆU", https://www.nguyencong.com/tong-quan-ve-khoi-rubik-3x3-va-cac-ky-hieu/ (Truy cập lần cuối ngày 15/5/2022)

[6] Quốc Hùng, "Sự ra đời của khối Rubik", https://khoahocphattrien.vn/kham-pha/su-ra-doi-cua-khoi-rubik/20200611102810635p1c879.htm (Truy cập lần cuối ngày 15/5/2022)

[7] Phạm Hải, "OpenGL là gì? OpenGL có tác dụng gì?", https://quantrimang.com/tim-hieu-ve-open-gl-157697 (Truy cập lần cuối ngày 15/5/2022)

[8] Admin https://dayhoc.pltpro.net, "Windows Forms App Bài 1: Làm quen với Windows Forms App (.Net Framework) bằng Visual Studio", https://dayhoc.pltpro.net/2020/05/lam-quen-voi-windows-forms-app-net-framework-bang-visual-studio/ (Truy cập lần cuối ngày 15/5/2022)

[9] http://bugnetproject.com, "Visual Studio là gì?", http://bugnetproject.com/visual-studio-la-gi/ (Truy cập lần cuối ngày 15/5/2022)

[10] Katharina Kurniasih, " The Amazing Benefits of Playing Rubik's Cubes For Our Brain ", https://4nids.com/2017/05/26/the-amazing-benefits-of-playing-rubiks-cubes-for-our-brain/ (Truy cập lần cuối ngày 15/5/2022)

[11] Wikipedia," Phương pháp CFOP ", https://vi.wikipedia.org/wiki/Phương_pháp_CFOP (Truy cập lần cuối ngày 15/5/2022)

[12] Wikipedia , "Lập phương Rubik", https://vi.wikipedia.org/wiki/Lập_phương_Rubik (Truy cập lần cuối ngày 15/5/2022)

[13] Kevin (Kevli) Li, "The Arcball", http://courses.cms.caltech.edu/cs171/assignments/hw3/hw3-notes/notes-hw3.html#:~:text=The%20Arcball%20is%20an%20intuitive,in%20a%203D%20computer%20environment (Truy cập lần cuối ngày 15/5/2022)