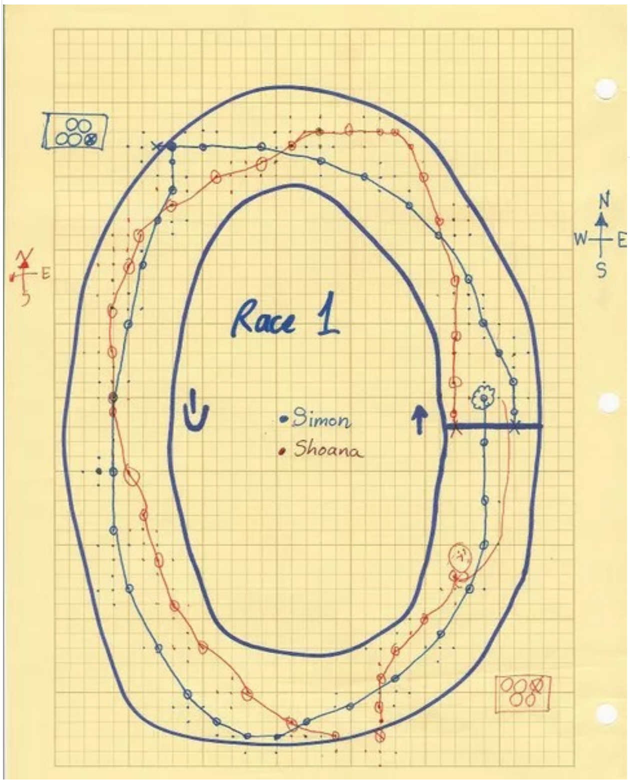
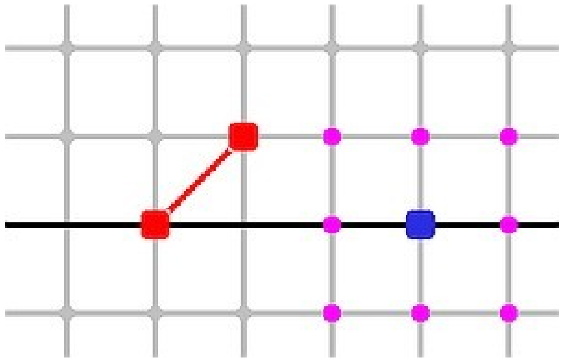
F1-RaceTrack (the game)

F1-RaceTrack is a game that uses classic (Newtonian) mechanics to simulate an F1 car race.

Martin Gardner, a science communicator famous for his mathematical puzzles, described the mathematical game RACETRACK in Scientific American in the early 1970s as a new game for the intellect (not luck) https://link.springer.com/chapter/10.1007/978-3-642-13122-6_26 It was played on a sheet of graph paper with colored pencils.



On a sheet of graph paper, a race track was drawn, and each player marked the initial location of their car at the starting line by drawing a point at the intersection of a grid.



Each car has a horizontal speed and a vertical speed (at the start of the game, both speeds are zero). To simulate acceleration, braking, and direction changes, on each turn, each player can change their horizontal speed by a maximum of one grid per turn, and can similarly change their vertical speed by a maximum of one grid per turn.

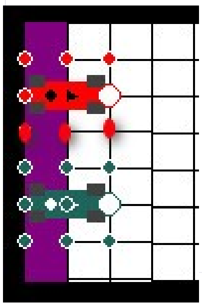
The winner is the first player to safely cross the finish line.

The game introduced concepts of speed, acceleration, and vector components.

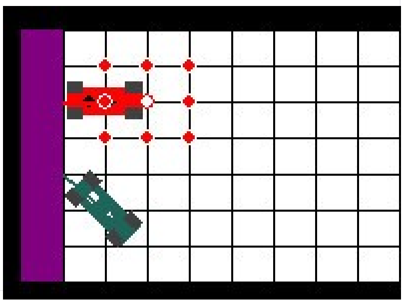
A good description of the original game is available at:

https://en.wikipedia.org/wiki/Racetrack_(game)

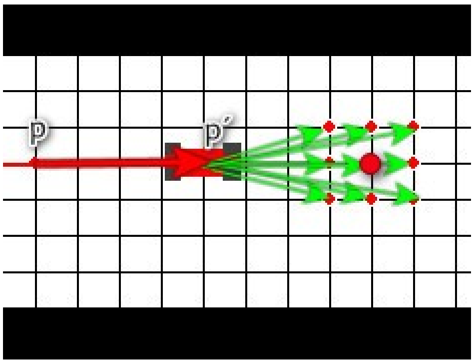
F1-RaceTrack is a computer game that simulates and recreates the original game, with Formula 1 cars.



Initially, the cars can move to any of the eight intersections surrounding them, though to establish logic, they must do so (at least at the start) in the direction of the track; in the graphic on the right, as we have moved one square, we will already have a speed, the red car has a horizontal speed Vh = +1

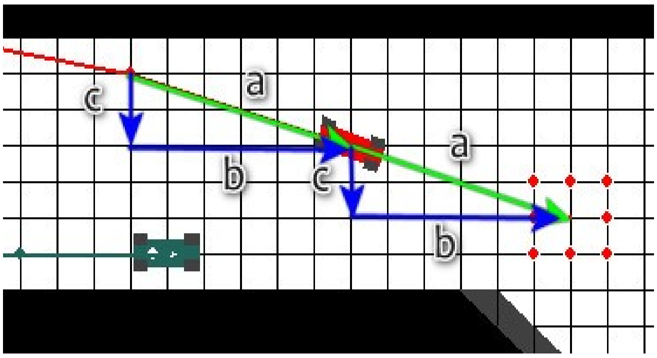


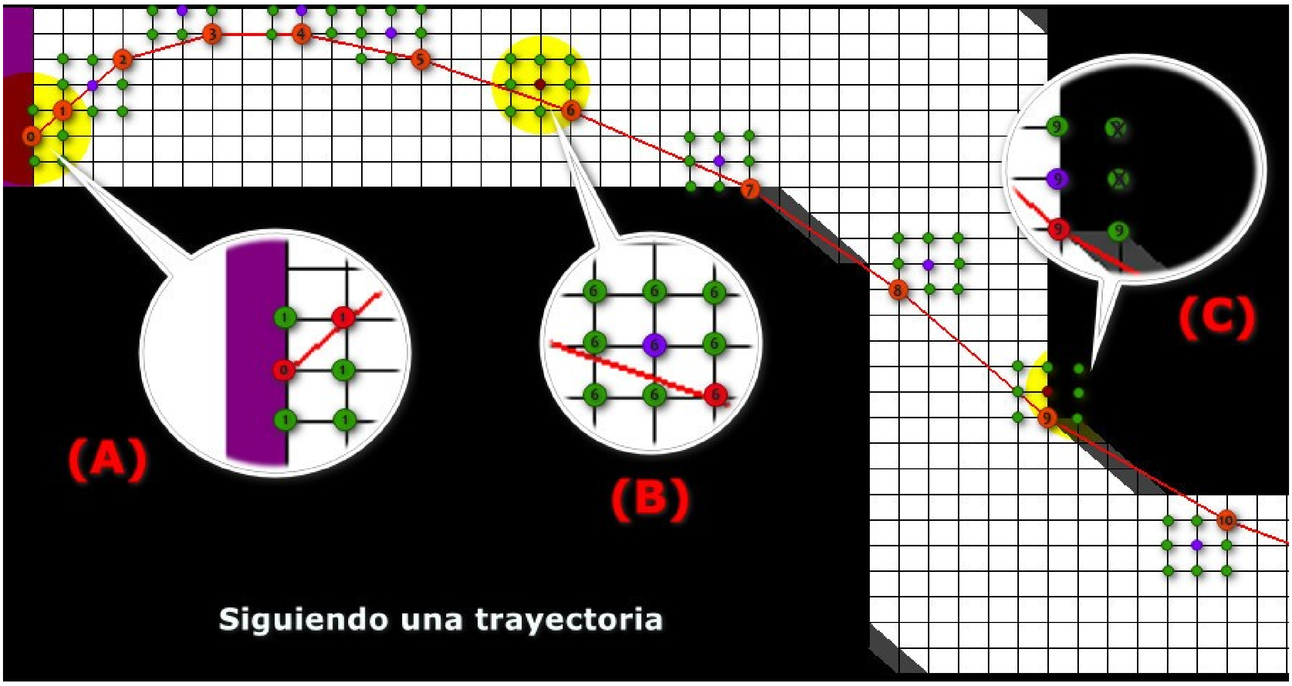
but its vertical speed Vv is = 0 since it has only moved horizontally; however, the green one has Vh = +1 and Vv = -1 since it has moved diagonally downward.



In subsequent movements, the point (red) that will serve as our reference to move our car, and that represents the position it would occupy if we make no changes and continue moving solely by inertia, is the projection of the vector **p** to **p′** from the previous movement. Following the rule that we can increase or decrease our horizontal and/or vertical speed by one square each turn, we can move to that point and all the intersections surrounding it, with which we simulate changes in speed and direction.

For example, if we are coming with a speed vector **a** (which we can decompose into a horizontal speed **b** and a vertical speed **c**), by maintaining our speed and direction, we would move like the red car in the figure. However, if we choose to change our speed and/or direction, we could move to any of the red points to brake, accelerate, and/or turn.





In the figure above, we start with the car located at the start line (point **0**) and have the first 10 points of a trajectory (points and red vectors connecting them). The red points are the chosen trajectory, the purple ones our reference points (obtained by projecting the previous trajectory), which represent the car's movement if we continued with the speeds (Vv and Vh) unchanged, and the green ones the possibilities of choice on each turn if we change by one unit (positive or negative) any of the speeds (Vh or Vv) surrounding our reference point (purple). To reach the finish line, we could follow many trajectories (as in a real race), some will be good, others not so much but at least will keep us on the track, others will take us off it, but there will always be at least one optimal trajectory that will take us to the finish line in the fewest possible movements. This is the case with the trajectory of red vectors in the figure; on each turn, we choose one of the possible points. In the first turn of the game, we can choose any of the intersections surrounding point **0**, since our Horizontal Speed (Vh) and Vertical Speed (Vv) are null.

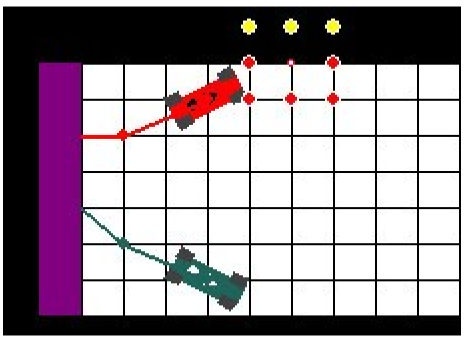
Let's look at some cases for better understanding:

Case **(A)**: We are at the start and choose on our first turn point **1**, increasing our Vv by one unit and also our Vh by one unit, (Vv = 1 Vh = 1), this will make our new reference point the first purple point, as if in the next turn (**2**) we wanted to maintain our speed (Vv and Vh) that would be the chosen one.

Case **(B)**: We are in turn **6**, in the previous turn we have moved from point **4** to **5** and as we had a Vh = 4 and a Vv = -1 our new reference point will be projecting that vector (purple point **6**). We choose to continue increasing our Vh by one unit so it will be Vh = 5 and continue increasing in negative our Vv which now will be: Vv = -2 occupying the red point **6**.

Case **(C)**: We are in turn **9**, we have maintained our Vh (Vh = 5) and have increased our Vv (Vv = -5), but we could have chosen any of the intersections surrounding the purple **9**, except those that are off the track (intersections of black squares). There will be intersections that even though they are within the track, in the next turn will inevitably take us off it. For example, if we had chosen the purple point **9** or the intersection just above it in turn **10**, we would almost certainly have gone off track. Choosing the optimal trajectory is as difficult as doing it with a car in a real race.

Opening the file: *Tutorial.html*, we can recreate the movement of the original game and use the mouse to move the car (black dot) through the grid intersections. The red dot represents the projection of our current velocity (initially zero) and shows where we will go if we maintain our vertical and horizontal velocity unchanged. The surrounding green intersections indicate where we can go by increasing (or decreasing) these velocities by one unit.



The goal od the game is to stay on the track and reach the finish line in the fewest number of moves, and if we are competing against another car, to arrive before it. Although the premise is very simple, we will need to follow a strategy similar to what we would employ if we were driving a car in real life, anticipating turns and moderating speed on curves to avoid going off the track, and then accelerating. And mastering this is only within the reach of the best. Do you dare to compete against Fernando Alonso?

The F1-Racetrack game streamlines and makes the race development very easy, and also shows us in the "replays" the recreated movement of the cars, surprising us with how much it resembles a real F1 race having used exclusively vectors to move the cars!

For a better understanding of the game's dynamics on the computer and to explore all their posibilities, read the document: F1-RaceTrack_Instructions.pdf