**Ogólne założenia działania AI w grze polegają na:**

1. Wybraniu pikseli wokół siebie w określonym promieniu
2. Na podstawie czystej geometrii wyznaczeniu dla każdego piksela wartości „zakrzywionej geometrii” (Nie wiemy jak to lepiej nazwać).
3. Przepuszczeniu tych wartości przez jakąś matematyczną funkcję (inną dla każdego bota), która zwróci wartość liczbową odpowiadającą opłacalności danego piksela.
4. Zapisaniu wartości opłacalności każdego zeskanowanego w tej klatce piksela.
5. Wybraniu tego piksela, który się najbardziej opłaca.
6. Ustawienie skrętu auta na taki kąt, pod jakim znaleźliśmy ten piksel (niech auto jedzie w stronę najbardziej opłacalnego piksela).

**„Zakrzywiona geometria”**

Piksele na torze w zakrzywionej geometrii również mają wartości (x,y) (Z perspektywy czasu można było to nazwać inaczej np. (w,y) ).

**Wartość x** rośnie wraz z przebytym dystansem wyścigu (czarna linia pokazuje wzrost wartości x). W przypadku zakrętu jest to liczone na podstawie zatoczonego kąta, a w przypadku prostej liczone jest to zwyczajnie jako normalna odległość od początku prostej (0~130).

**Obraz zawierający zrzut ekranu, linia, Równolegle, Wielobarwność

Zawartość wygenerowana przez sztuczną inteligencję może być niepoprawna.Obraz zawierający zegar, krąg, Grafika, kreskówka

Zawartość wygenerowana przez sztuczną inteligencję może być niepoprawna.Wartość y** oznacza wychylenie od środka toru. Punkty idealnie na środku mają wartość zero, punkty po lewej stronie względem kierunku jazdy mają wartości ujemne, a po prawej stronie wartości dodatnie (-65~65) (żółta linia pokazuje jak rośnie y). W przypadku zakrętu wartością jest odległość od środka koła minus 65 (szerokość wynosi 130, dlatego odejmujemy 65, żeby wartość na środku wyszła zero). Jeśli zakręcamy w prawo, to trzeba odwrócić wartości, żeby po lewej były ujemne.

**Np. Czerwony punkt ma wartość około (120,-30), a niebieski około (25,40).**

**Dokładny opis:**

1. Każdy kafelek toru ma na początku gry liczone wartości zakrzywionej geometrii dla każdego piksela (130x130), które są zapisywane na dwuwymiarowej liście. Indeksami pod którymi się to zapisuje jest relatywna pozycja danego piksela względem początku kafelka. Np. values[0][0] oznacza wartości (x,y) zakrzywionej geometrii dla piksela, który leży w lewym górnym rogu ekranu.
2. Podczas „skanu” wybieramy po kolei kąty od -180 do 180 i patrzymy jaki piksel znajduje się pod takim kątem od samochodu w konkretnej odległości (promieniu r). Z użyciem trygonometrii liczone są „normalne” koordynaty piksela na planszy.
3. Następnie odejmujemy od tego koordynaty kafelka toru, żeby uzyskać relatywną pozycję szukanego piksela względem pozycji kafelka. Uzyskane relatywne koordynaty będą się już mieściły w szerokości i wysokości kafelka (tzn. będą z przedziału od 0 do 130).

Nie pozostaje już wtedy nic innego jak odwołać się do dwuwymiarowej listy podając za indeksy relatywne koordynaty i lista zwróci nam gotowe wartości (x,y) dla zakrzywionej geometrii (tzn. Dowiemy się jaką drogę trzeba przebyć do tego piksela od początku kafelka oraz jak bardzo wychylony od środka kafelka jest ten piksel).

1. Tak wyznaczone wartości zakrzywionej geometrii przepuszczamy przez matematyczną funkcję (różną dla każdego bota), żeby zobaczyć jak bardzo w stronę danego piksela opłaca się jechać (im większa wartościowość tym bardziej się opłaca).

4,5,6. Zbieramy w słowniku wartościowości wszystkich zeskanowanych pikseli pamiętając dla jakiego kąta się one pojawiły. Wybieramy największą wartościowość. Patrzymy dla jakiego kąta wystąpiła i na końcu ustawiamy botowi skręt kierownicy właśnie na ten kąt.