Arkadiusz Sadowski
Sebastian Sarnecki
Jędrzej Smaruj

Sztuczna Inteligencja Projekt

Aplikacja uruchamiająca odpowiedni funkcjonalności telefonu w zależności od wykonanego ruchu

Opis problemu

Aplikacja działająca pod systemem Android, która przy pomocy akcelerometru wbudowanego w telefon wykrywa kształt przekazany za pomocą ruch urządzeniem. Wyryty ruch przyporządkowuje do wcześniej nauczonych kształtów i uruchamia na tej podstawie odpowiednią funkcjonalność telefonu.

Opis użytej metody

Do wykrycia, nauki oraz porównania kształtów wykonywanych telefonem ruchów z czytywanych przy pomocy akcelerometru użyliśmy biblioteki **$1 Recognizer**. Biblioteka przeznaczona jest dla nie doświadczonych jeszcze programistów, w szczególności studentów. Jej celem jest umożliwienie w przystępny i krótki sposób implementacji operacji na ruchach w urządzeniach mobilnych. Pomimo swojej prostoty metoda wykazuje wysoką skuteczność, porównywalną do innych bardziej skomplikowanych metod. Z tych względów zdecydowaliśmy się użyć właśnie $1 Recognizer.

Algorytm wykorzystywany w $1 Recognizer opiera się na czterech głównych krokach: przeskalowanie wczytanych punktów, rotacja figury, skalowanie oraz znalezienie optymalnego ustawienia by uzyskać najwyższy wynik punktowy dla figur, które przyporządkowujemy do wzorów. Zanim zaczniemy przyrównywać ruchy do wzorców należy nauczyć aplikację jak one wyglądają.

Przeskalowywanie

Aby łatwiej było porównywać kształty M (które dla różnych wejść może być różne) wczytanych punktów przeskalowywane jest na N (teraz już zawsze takie same) równoodległych punktów. N zostało dobrane tak aby nie utracić dokładności lecz za razem nie powiększać czasu porównywania. Twórcy biblioteki w praktyce zauważyli, że N=64 działa optymalnie.

Rotacja

Po przeskalowaniu otrzymany kształt jest obracany aby znaleźć najbardziej optymalną pozycję do porównywania z danym wzorcem. Obracanie to mogło by zostać wykonane metodą brutalną. Jednak można je łatwo przyspieszyć. Wystarczy zdefiniować kąt pomiędzy pierwszym punktem (z którego rozpoczął się ruch) a środkiem figury. Następnie rotować figurę tak aby ten kont wynosił 0.

Transponowanie do kwadratu

Następnie ze względu na to, że wszystkie kształty mogą być różnego rozmiaru transponujemy je na takie, które wpisane są w kwadrat o określonej wielkości.

Znalezienie optymalnego kąta dla maksymalnego wyniku

Do tego momentu zarówno wzorce jak i ruchy do rozpoznania były traktowane tak samo. Ten krok natomiast jest przeznaczony już tylko dla ruchów, które chcemy rozpoznać. Najpierw przy pomocy poniższego wzoru porównujemy kandydata C z każdym zapamiętanym wzorcem T_i by znaleźć średni dystans pomiędzy odpowiadającymi sobie punktami d_i.

$$d_i = \frac{\sum_{k=1}^{N} \sqrt{(C[k]_x - T_i[k]_x)^2 + (C[k]_y - T_i[k]_y)^2}}{N}$$

Wzór T_i z najmniejszą odległością d_i do kandydata C jest wynikiem rozpoznawania. Minimalna odległość d_i jest konwertowany do wyniku z zakresu [0-1] przy pomocy następującego wzoru:

$$score = 1 - \frac{d_i^*}{\frac{1}{2}\sqrt{size^2 + size^2}}$$

Size to długość boku kwadratu do którego transponowaliśmy figurę.

Opis realizacji zadania

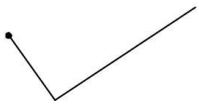
Wpierw zapoznaliśmy się z opisem biblioteki $1 Recognizer. Ze względu na fakt, że biblioteka działa na zasadach open source zapoznaliśmy się z implementacja biblioteki. Przy pomocy tej biblioteki zaimplementowaliśmy aplikację.

Zdecydowaliśmy, że wykrywanie ruchu rozpocznie się w momencie przyciśnięcia przycisku widocznego na ekranie. Zakończymy je w momencie ponownego naciśnięcia tego przycisku.

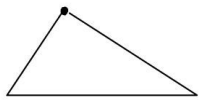
Najpierw uczymy aplikację jak wyglądają poszczególne wzorce. Oznacza to, że po zakończeniu wykrywania ruchu, wczytana figura jest skalowana, rotowana, transponowana i zapamiętywana.

Następnie przy porównywaniu po zakończeniu wykrywania ruchu, wczytana figura jest skalowana, rotowana, transponowana i porównywana do wszystkich zapamiętanych wzorców. Po przyporządkowaniu do danego wzorca wywoływana jest odpowiednia funkcjonalność.

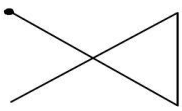
Kształty i funkcjonalności wywoływane przy ich wykryciu



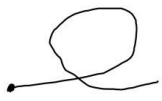
Włącza i wyłącza latarkę.



Włącza i wyłącza wifi.



Włącza i wyłącza tryb cichy.



Robi screenshot.

Prezentacja osiągniętych wyników ich dyskusja

Nauczanie i testy zostały przeprowadzone w następujący sposób. Trzy osoby rozpoczynając od nie nauczonej aplikacji uczyły ją w trzech fazach. W każdej z nich uczyły każdego kształtu trzy razy w zdefiniowany wcześniej sposób (wyjaśnione poniżej). Po każdej fazie wykonywano za każdym razem pięć takich samych testów (także zaprezentowane poniżej) na każdy kształt.

Fazy nauki:

I faza	II faza	III faza
mały wolny ruch	mały wolny ruch	mały wolny ruch
mały wolny ruch	duży szybki ruch	spłaszczony szybki ruch
duży wolny ruch	spłaszczony wolny ruch	bardzo duży ruch

Testy po każdej fazie:

1. mały wolny ruch
2. mały wolny ruch
3. duży wolny ruch
4. mały szybki ruch
5. spłaszczony wolny ruch

Uzyskane wyniki:

Pierwsza osoba

Po pierwszej fazie nauki					
Check	Rozpoznano poprawnie	Nie rozpoznał żadnego wzoru	Rozpoznano poprawnie	Rozpoznano poprawnie	Nie rozpoznał żadnego wzoru
X	Nie rozpoznał żadnego wzoru	Rozpoznano poprawnie	Rozpoznano poprawnie	Rozpoznano poprawnie	Nie rozpoznał żadnego wzoru
Triangle	Rozpoznano poprawnie	Rozpoznano poprawnie	Nie rozpoznał żadnego wzoru	Rozpoznano poprawnie	Wykryto inny wzrór
Pig tail	Rozpoznano poprawnie	Rozpoznano poprawnie	Nie rozpoznał żadnego wzoru	Rozpoznano poprawnie	Nie rozpoznał żadnego wzoru
Po drugiej fazie nauki					
Check	Rozpoznano poprawnie	Rozpoznano poprawnie	Rozpoznano poprawnie	Rozpoznano poprawnie	Rozpoznano poprawnie
X	Rozpoznano poprawnie	Rozpoznano poprawnie	Rozpoznano poprawnie	Rozpoznano poprawnie	Rozpoznano poprawnie
Triangle	Rozpoznano poprawnie	Nie rozpoznał żadnego wzoru	Rozpoznano poprawnie	Rozpoznano poprawnie	Nie rozpoznał żadnego wzoru

Pig tail	Rozpoznano poprawnie	Rozpoznano poprawnie	Rozpoznano poprawnie	Rozpoznano poprawnie	Rozpoznano poprawnie
Po trzeciej fazie nauki					
Check	Rozpoznano poprawnie	Rozpoznano poprawnie	Rozpoznano poprawnie	Wykryto inny wzrór	Rozpoznano poprawnie
X	Rozpoznano poprawnie	Wykryto inny wzrór	Rozpoznano poprawnie	Rozpoznano poprawnie	Rozpoznano poprawnie
Triangle	Rozpoznano poprawnie	Rozpoznano poprawnie	Rozpoznano poprawnie	Rozpoznano poprawnie	Wykryto inny wzrór
Pig tail	Rozpoznano poprawnie	Rozpoznano poprawnie	Rozpoznano poprawnie	Rozpoznano poprawnie	Rozpoznano poprawnie

Druga osoba

Po pierwszej fazie nauki					
Check	Nie rozpoznał żadnego wzoru	Rozpoznano poprawnie	Rozpoznano poprawnie	Rozpoznano poprawnie	Nie rozpoznał żadnego wzoru
X	Nie rozpoznał żadnego wzoru	Wykryto inny wzrór	Rozpoznano poprawnie	Nie rozpoznał żadnego wzoru	Nie rozpoznał żadnego wzoru
Triangle	Rozpoznano poprawnie	Rozpoznano poprawnie	Nie rozpoznał żadnego wzoru	Rozpoznano poprawnie	Rozpoznano poprawnie
Pig tail	Nie rozpoznał żadnego wzoru	Rozpoznano poprawnie	Nie rozpoznał żadnego wzoru	Rozpoznano poprawnie	Rozpoznano poprawnie
Po drugiej fazie nauki					
Check	Nie rozpoznał żadnego wzoru	Rozpoznano poprawnie	Wykryto inny wzrór	Rozpoznano poprawnie	Nie rozpoznał żadnego wzoru
X	Wykryto inny wzrór	Rozpoznano poprawnie	Rozpoznano poprawnie	Rozpoznano poprawnie	Wykryto inny wzrór
Triangle	Wykryto inny wzrór	Rozpoznano poprawnie	Rozpoznano poprawnie	Rozpoznano poprawnie	Rozpoznano poprawnie
Pig tail	Rozpoznano poprawnie	Rozpoznano poprawnie	Rozpoznano poprawnie	Wykryto inny wzrór	Rozpoznano poprawnie
Po trzeciej fazie nauki					
Check	Rozpoznano poprawnie	Rozpoznano poprawnie	Rozpoznano poprawnie	Rozpoznano poprawnie	Rozpoznano poprawnie

X	Rozpoznano poprawnie	Rozpoznano poprawnie	Rozpoznano poprawnie	Rozpoznano poprawnie	Rozpoznano poprawnie
Triangle	Rozpoznano poprawnie	Rozpoznano poprawnie	Rozpoznano poprawnie	Rozpoznano poprawnie	Rozpoznano poprawnie
Pig tail	Rozpoznano poprawnie	Rozpoznano poprawnie	Rozpoznano poprawnie	Rozpoznano poprawnie	Rozpoznano poprawnie

Trzecia osoba

Po pierwszej fazie nauki					
Check	Nie rozpoznał żadnego wzoru	Nie rozpoznał żadnego wzoru	Nie rozpoznał żadnego wzoru	Rozpoznano poprawnie	Nie rozpoznał żadnego wzoru
X	Nie rozpoznał żadnego wzoru	Nie rozpoznał żadnego wzoru	Rozpoznano poprawnie	Rozpoznano poprawnie	Nie rozpoznał żadnego wzoru
Triangle	Rozpoznano poprawnie	Nie rozpoznał żadnego wzoru	Nie rozpoznał żadnego wzoru	Rozpoznano poprawnie	Nie rozpoznał żadnego wzoru
Pig tail	Nie rozpoznał żadnego wzoru	Rozpoznano poprawnie	Rozpoznano poprawnie	Rozpoznano poprawnie	Rozpoznano poprawnie
Po drugiej fazie nauki					
Check	Wykryto inny wzrór	Rozpoznano poprawnie	Rozpoznano poprawnie	Rozpoznano poprawnie	Nie rozpoznał żadnego wzoru
X	Rozpoznano poprawnie	Wykryto inny wzrór	Rozpoznano poprawnie	Wykryto inny wzrór	Rozpoznano poprawnie
Triangle	Rozpoznano poprawnie	Rozpoznano poprawnie	Rozpoznano poprawnie	Wykryto inny wzrór	Nie rozpoznał żadnego wzoru
Pig tail	Rozpoznano poprawnie	Rozpoznano poprawnie	Rozpoznano poprawnie	Rozpoznano poprawnie	Rozpoznano poprawnie
Po trzeciej fazie nauki					
Check	Rozpoznano poprawnie	Rozpoznano poprawnie	Rozpoznano poprawnie	Rozpoznano poprawnie	Rozpoznano poprawnie
X	Rozpoznano poprawnie	Rozpoznano poprawnie	Rozpoznano poprawnie	Rozpoznano poprawnie	Rozpoznano poprawnie
Triangle	Rozpoznano poprawnie	Rozpoznano poprawnie	Rozpoznano poprawnie	Rozpoznano poprawnie	Rozpoznano poprawnie
Pig tail	Rozpoznano poprawnie	Rozpoznano poprawnie	Rozpoznano poprawnie	Rozpoznano poprawnie	Rozpoznano poprawnie

Podsumowanie wyników

Po pierwszej fazie skuteczność wynosi 55% jednak bardzo niski na poziomie 3% jest błąd w wykryciu wzoru. Po drugiej fazie skuteczność wzrasta do 75% ale do 15% wrasta pomyłka. Po trzeciej fazie skuteczność wynosi już 98% z 2% wykryciem innego niż zadany ruch.

Wnioski:

Wyraźnie widać, że przy coraz większej ilości wgranych wzorów zwiększa się skuteczność rozpoznania. Przy już 3 wgranych wzorach dla każdego ruchu skuteczność wynosi ponad 50% a przy 9 aż 98% co uważamy za bardzo zadowalające.

Ciekawym zjawiskiem jest wzrost błędnych rozpoznań przy 6 wgranych wzorach. Podejrzewamy, że wynika on z niedokładnie wgranych wzorców. Ruchy takie jak “duży szybki” i “spłaszczony” mogą niedokładnie wykonane przypominać inne ruchy co powoduje błędną interpretację przez aplikację. Jednak przy podaniu większej liczby wzorców problem staje się rozwiązany.

Zaobserwowaliśmy, że ruch typu “pig tail”, który wyraźnie różni się od pozostałych wzorców ze względu na swoją kolistość, jest wykrywany z wysoką skutecznością już przy mniejszej liczbie ruchów. Pokazuje to, że w przypadku wybrania wyraźnie różnych wzorców wystarczyło by wgranie mniejszej liczby przykładów do osiągnięcia podobnej skuteczności.