Notatki do obrony

Slajd 1 – STRONA TYTUŁOWA

Dzień dobry, nazywam się Piotr Noga. Dzisiaj przedstawię moją pracę inżynierską pod tytułem „System do wykrywania i rozpoznawania obiektów na obrazach z kamery samochodowej z wykorzystaniem symulatora CARLA”. Praca była realizowana na Wydziale Informatyki, Elektrotechniki i Automatyki Uniwersytetu Zielonogórskiego, pod opieką pana dr hab. inż. Marka Kowala, prof. UZ. Prezentacja potrwa około 6–8 minut i będzie składała się z krótkiego wprowadzenia, omówienia celu i zakresu pracy, wykorzystanych technologii, przeprowadzonych eksperymentów oraz najważniejszych wyników i wniosków.

Slajd 2 – AGENDA

Agenda mojej prezentacji wygląda następująco. Najpierw przedstawię problem i motywację stojącą za tą pracą – skąd w ogóle wziął się pomysł na taki temat. Następnie omówię cel i zakres pracy, czyli co dokładnie zostało zrealizowane. Kolejna część to wykorzystane technologie – krótko opiszę środowisko CARLA, YOLOv4 i narzędzia, których użyłem. Potem przejdę do dwóch eksperymentów, które przeprowadziłem oraz problemów, na które natrafiłem. Na końcu pokażę najważniejsze wyniki i wnioski z pracy.

Slajd 3 – PROBLEM I MOTYWACJA

Na początku kilka słów o problemie i motywacji. Po pierwsze, rośnie rola systemów wspomagania kierowcy, czyli ADAS (Advanced Driver Assistance Systems), oraz pojazdów autonomicznych. Jednym z kluczowych elementów takich systemów jest szybka i dokładna detekcja obiektów z kamery – na przykład samochodów, pieszych, znaków drogowych czy sygnalizacji świetlnej – w czasie rzeczywistym. Bez tego warstwy odpowiedzialne za podejmowanie decyzji i sterowanie pojazdem nie mają wiarygodnych danych wejściowych. Po drugie, testowanie algorytmów detekcji tylko w prawdziwym ruchu drogowym jest kosztowne, czasochłonne i wiąże się z ryzykiem. Trudno też powtarzalnie odwzorować dokładnie te same scenariusze, warunki pogodowe i oświetleniowe. Dlatego potrzebne jest bezpieczne, powtarzalne środowisko symulacyjne, takie jak CARLA, które umożliwia generowanie realistycznych scen z ruchu drogowego. Po trzecie, pojawia się złożony problem techniczny – symulator przechowuje położenia obiektów w przestrzeni trójwymiarowej, natomiast algorytmy detekcji działają na obrazach dwuwymiarowych. Trzeba więc przejść od 3D do 2D, wyznaczyć bounding boxy CARLA w obrazie kamery i dopiero wtedy porównać je z detekcjami YOLOv4 za pomocą miary IoU. To przejście i poprawne zgranie obu źródeł informacji było jednym z większych wyzwań.

Slajd 4 – CEL I ZAKRES PRACY

Celem pracy było opracowanie systemu, który umożliwia wykrywanie i rozpoznawanie obiektów na obrazach pochodzących z kamery samochodowej w różnych warunkach pogodowych. Skupiłem się na czterech najważniejszych klasach obiektów z punktu widzenia ruchu drogowego: samochodach, ludziach, znakach drogowych oraz sygnalizacji świetlnej. Zakres pracy obejmował kilka etapów. Pierwszy etap to zapoznanie się ze środowiskiem symulacji jazdy samochodem CARLA – jego możliwościami, sposobem konfiguracji scen, kamer i czujników. Drugi etap stanowił przegląd metod wykrywania obiektów na obrazach z kamery samochodowej, ze szczególnym uwzględnieniem rodziny algorytmów YOLO. W kolejnym kroku zaprojektowałem i wdrożyłem system, który łączy symulator CARLA z modelem YOLOv4 i umożliwia zarówno detekcję online, jak i analizę offline. Następnie przeprowadziłem testy weryfikujące skuteczność zaprojektowanego systemu, w różnych scenariuszach i warunkach. Na końcu opracowałem wnioski i możliwe kierunki dalszego rozwoju.

Slajd 5 – WYKORZYSTANE TECHNOLOGIE

W pracy wykorzystałem kilka kluczowych technologii. Pierwszą z nich jest CARLA – symulator opuen-source zaprojektowany specjalnie do badań nad autonomiczną jazdą i systemami wspomagania kierowcy. CARLA pozwala tworzyć szczegółowe scenariusze ruchu drogowego, definiować trajektorie pojazdów, dodawać pieszych, zmieniać warunki pogodowe i porę dnia oraz podłączać wirtualne czujniki, takie jak kamery. Drugą technologią jest język Python, który wykorzystałem do integracji poszczególnych elementów systemu – komunikacji z symulatorem, przetwarzania danych z kamer oraz uruchamiania modeli detekcji obiektów. Trzecim elementem jest YOLOv4 – szybki algorytm detekcji obiektów w czasie rzeczywistym. W swojej pracy korzystałem zarówno z pełnego modelu YOLOv4, zapewniającego lepszą dokładność, jak i z lżejszej wersji YOLOv4 Tiny, która jest mniej wymagająca obliczeniowo, ale kosztem jakości detekcji. Dodatkowo korzystałem z biblioteki Ultralytics, która upraszcza uruchamianie i konfigurację modeli YOLO w trybie offline, co było szczególnie istotne przy drugim eksperymencie.

Slajd 6 – EKSPERYMENTY I PROBLEMY (przegląd)

W pracy przeprowadziłem dwa główne eksperymenty. Pierwszy dotyczył rozpoznawania i wykrywania obiektów w czasie rzeczywistym, bezpośrednio w symulatorze CARLA. Drugi eksperyment miał na celu dokładne sprawdzenie miary jakości detekcji poprzez porównanie wyników YOLO z ground truth wygenerowanym przez symulator, ale już w trybie offline. Po drodze pojawiło się kilka istotnych problemów. W warunkach online, pomimo użycia modelu Tiny, liczba klatek na

sekundę była bardzo niska – szczególnie przy ograniczonych zasobach CPU i GPU. To wprost wpływało na jakość i stabilność detekcji. Dodatkowo pojawiały się rozbieżności pomiędzy bounding boxami z CARLI a tymi generowanymi przez YOLO, co utrudniało rzetelne porównanie. Rozwiązaniem okazało się przeniesienie części analiz do trybu offline, zapisanie obrazów oraz ground truth, a następnie przeliczenie miary IoU dla poszczególnych scenariuszy bez ograniczeń wynikających z czasu rzeczywistego.

Slajd 7 – EKSPERYMENT 1 – ONLINE (obrazki YOLOv4 vs Tiny)

Na tym slajdzie pokazuję wyniki pierwszego eksperymentu, czyli detekcji w czasie rzeczywistym bezpośrednio w symulatorze. Po lewej stronie widać przykładowy obraz z detekcją modelu YOLOv4, a po prawej detekcję modelu YOLOv4 Tiny. W obu przypadkach mamy scenę z ruchem drogowym – pojazdy, pieszych, sygnalizację świetlną czy znaki. W przypadku pełnego modelu YOLOv4 detekcje są bardziej kompletne – widać większą liczbę poprawnie wykrytych obiektów, a bounding boxy lepiej pokrywają obiekty w scenie. Z kolei model Tiny działa szybciej i jest mniej obciążający dla sprzętu, ale skutkuje to częściową utratą dokładności – niektóre obiekty są pomijane albo detekcje są mniej precyzyjne. Kluczowym problemem w tym eksperymencie okazała się jednak nie tylko sama dokładność modelu, ale również liczba klatek na sekundę. Przy ograniczonych zasobach sprzętowych FPS spadał na tyle, że system nie nadawał się do stabilnej oceny jakości detekcji w warunkach real-time, co skłoniło mnie do przygotowania drugiego, offline'owego eksperymentu.

Slajd 8 – EKSPERYMENT 2 – OFFLINE (ground truth vs YOLO)

Drugi eksperyment został przeniesiony do trybu offline po to, aby móc w sposób wiarygodny ocenić jakość detekcji niezależnie od liczby klatek na sekundę. Na tym slajdzie po lewej stronie widzimy obraz z nałożonym ground truth z symulatora CARLA – są to referencyjne ramki obejmujące obiekty w scenie, których położenie i rozmiar są dokładnie znane z modelu 3D. Po prawej stronie znajduje się ten sam obraz, ale z nałożonymi wynikami detekcji YOLO. Dla każdej pary bounding boxów – jednego z ground truth i jednego z YOLO – obliczana była miara IoU, czyli Intersection over Union. Dzięki temu mogłem policzyć średnią IoU dla różnych scenariuszy i warunków: na przykład dla dobrej widoczności w dzień, dla deszczu czy dla scen nocnych. Podejście offline pozwoliło na odseparowanie problemu wydajności sprzętowej od samej jakości algorytmu detekcji. Zamiast walczyć z FPS-ami, mogłem skupić się na tym, jak dobrze model YOLOv4 odwzorowuje faktyczne położenie obiektów w obrazie, korzystając z precyzyjnego ground truth dostarczanego przez CARLĘ.

Slajd 9 – WYNIKI I WNIOSKI

Na podstawie przeprowadzonych eksperymentów można sformułować kilka najważniejszych wniosków. Po pierwsze, YOLOv4 rzeczywiście umożliwia detekcję obiektów w czasie rzeczywistym, ale w praktyce wymaga to bardzo wydajnych zasobów sprzętowych – zarówno CPU, jak i GPU. W warunkach ograniczonych zasobów lepszym podejściem okazało się przeniesienie analizy do trybu offline. Po drugie, analiza miary IoU pokazała wyraźnie, że najlepsze wyniki detekcji uzyskujemy dla obiektów znajdujących się bliżej kamery oraz przy dobrej widoczności, na przykład w dzień i bez opadów. Dla obiektów odległych oraz w trudnych warunkach pogodowych, takich jak deszcz czy noc, jakość detekcji wyraźnie się pogarsza. Po trzecie, zbudowany system umożliwia szczegółową analizę błędów – można dokładnie zobaczyć, w jakich sytuacjach YOLO się myli, czego nie wykrywa lub jak przesunięte są ramki detekcji względem ground truth. Dzięki temu system stanowi dobrą bazę do dalszych badań i rozwoju systemów autonomicznych – można na przykład testować inne modele detekcji, dodatkowe czujniki czy bardziej złożone scenariusze ruchu.

Slajd 10 – DZIĘKUJĘ ZA UWAGĘ

To już wszystko z mojej strony. Dziękuję Państwu za uwagę. Chętnie odpowiem teraz na ewentualne pytania dotyczące zarówno samego systemu, jak i wykorzystania symulatora CARLA czy przeprowadzonych eksperymentów.