In dit rapport zal ik niet alles vermelden wat er tijdens de projectweek is gedaan, maar alleen wat er bij herkansing is gebeurd. Ik wil beginnen met de feedback die ik van de leraren heb gekregen. Kort gezegd bestaat het uit drie punten:

1)Pas de abstractie en herbruikbaarheid van je algoritme aan.

2) In plaats van alles alleen te implementeren voor het huidige doel, probeer te bepalen hoe je dat abstracter zou kunnen (zoeken naar een reeks van n nummers, in plaats van zoeken naar 4 nummers).

3)Mogelijke uitbreidingen: bijv. preprocessing (bepalen aan de hand van image processing welk gebied uitgesneden moet worden om het adres te vinden, of bron bestand roteren).

Met betrekking tot de eerste twee punten. De code waardoor deze klachten werden ontvangen, was gekoppeld aan mijn algoritme, dat uit de gegeven tekst het adres kon selecteren dat we nodig hadden (ook handmatig gegeven). Het was behoorlijk zorgeloos geschreven en had een eenzijdige toepassing. Zijn onvoorzichtigheid was omdat de code te lang was daardoor ook moeilijk te lezen was en ook veel regels code bevatte die vermeden hadden kunnen worden. In plaats van dat algoritme te wijzigen, schreef ik een nieuw. Voor de duidelijkheid, het nieuwe algoritme kostte bijna 7 regels om het hele algoritme te vullen, in plaats van 63. Bovendien heeft het geen argument nodig in de vorm van een postcode. Nu wordt in plaats van de postcode de nieuwe functie gebruikt die door mij is geschreven check\_whether\_int\_in\_the\_row, die controleert of de gevonden tekst het adres is. Ze doet dit door te controleren op de aanwezigheid van 4 cijfers in de gevonden tekst. De genoemde functie kan niet alleen nuttig zijn in mijn taak, maar ook in andere. Het is niet beperkt tot het vinden van vier cijfers. De stappen om een adres te zoeken waren verrassend eenvoudig en hadden tegelijkertijd ook nadelen. Het grootste nadeel is bijvoorbeeld dat het alleen werkt met bepaalde tekstpatroon, waar het adres aan het einde staat. En als je hem iets anders geeft, zal hij hem waarschijnlijk niet kunnen vinden. Maar in tegenstelling hiermee schreef ik ook een ander deel van de code in hetzelfde algoritme, dat, ongeacht de tekst die eraan wordt gegeven, de gewenste combinatie van de opgegeven tekens vindt en deze samen met een indicatie van hun locatie retourneert.

Wat betreft de mogelijke uitbreiding, besloot ik de aanbeveling op te volgen. Als gevolg hiervan maakte ik een algoritme dat de vensters in de brief kon vinden en ze daarna kon knippen, waardoor alleen het vermelde venster overblijft. De code is weer geschreven voor een enkele taak en wordt niet waarschijnlijker gebruikt in veel andere taken. Het hangt allereerst samen met zijn methode. Het was de bedoeling dat hij naar slechts 1 rechthoek zou zoeken en niets anders. Ook kan hij dit venster alleen vinden als hij al een bewerkte foto op voorraad heeft. Wat betreft het aantal rechthoeken, het kan vrij snel en gemakkelijk wordt veranderd, maar met betrekking tot de rest treden er al problemen op. Als in het kort het bovenstaande algoritme uit de volgende stappen bestaat: Het is van links naar rechts, regel voor regel controleert elke pixel op de aanwezigheid van een zwarte pixel. Nadat de eerste zwarte pixel is gevonden, worden de coördinaten van de genoemde pixel opgeslagen. Vervolgens kijkt hij naar de volgende pixels of deze zwart zijn. Als de cursor na het vinden van een combinatie van zwarte tekens achtereenvolgens wit wordt, controleert deze of de lengte van de zwarte pixels in een rij overeenkomt met de opgegeven minimale lengte van de rechthoek. Zo ja, dan begint hij te controleren op de aanwezigheid van zwarte kleur die al in de hoogte is. Nadat hij de hoogte heeft gevonden, controleert hij opnieuw of deze voldoet aan de minimale hoogte. Zo ja, dan worden de coördinaten van de linkerbovenhoek en de rechteronderhoek van de rechthoek opgeslagen. Deze coördinaten worden vervolgens gebruikt in de functie voor het bijsnijden van foto's.