Gépi látás

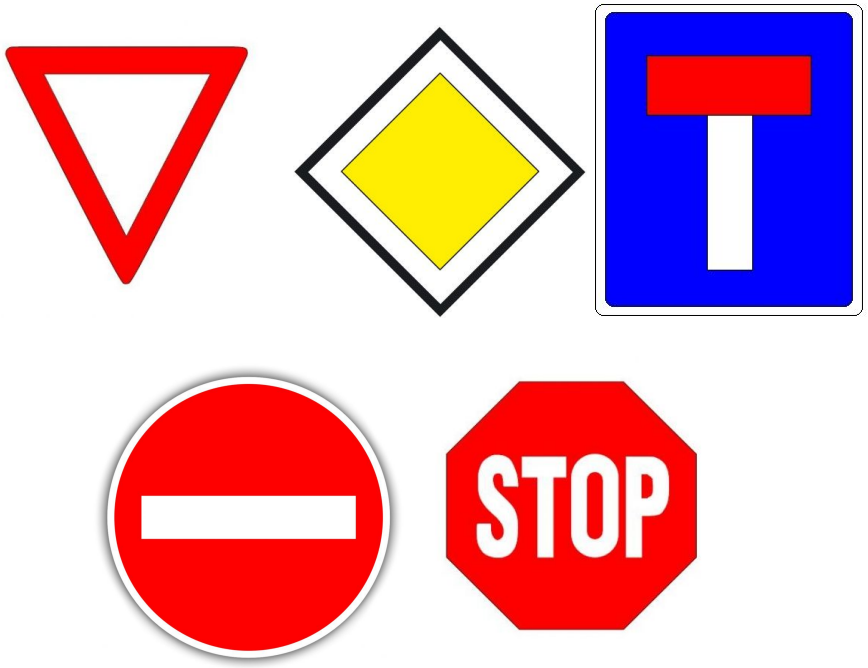
Közlekedési tábla felismerése

Tartalomjegyzék

Bevezetés, megoldandó feladat kifejtése

A megoldandó feladat 5 előre definiált közlekedési tábla felismerése a gépi látás módszereinek segítségével. A választott táblák nevei és kinézetei a következők:

- elsőbbségadás kötelező
- főútvonal
- zsákutca
- behajtani tilos
- STOP tábla



Megoldáshoz szükséges elméleti háttér kifejtése

A gépi látás használata során olyan algoritmusokat használunk, amelyek segítenek digitális képeket értelmezni. A digitális képi bemenetből kimenetként egy leírás jön létre.

A képen végzett transzformációkhoz gyakran nem szükséges színformáció, így ezen “felesleges” adatok a szürkeárnyalatos képpé alakítás során eldobásra kerülnek. [2]

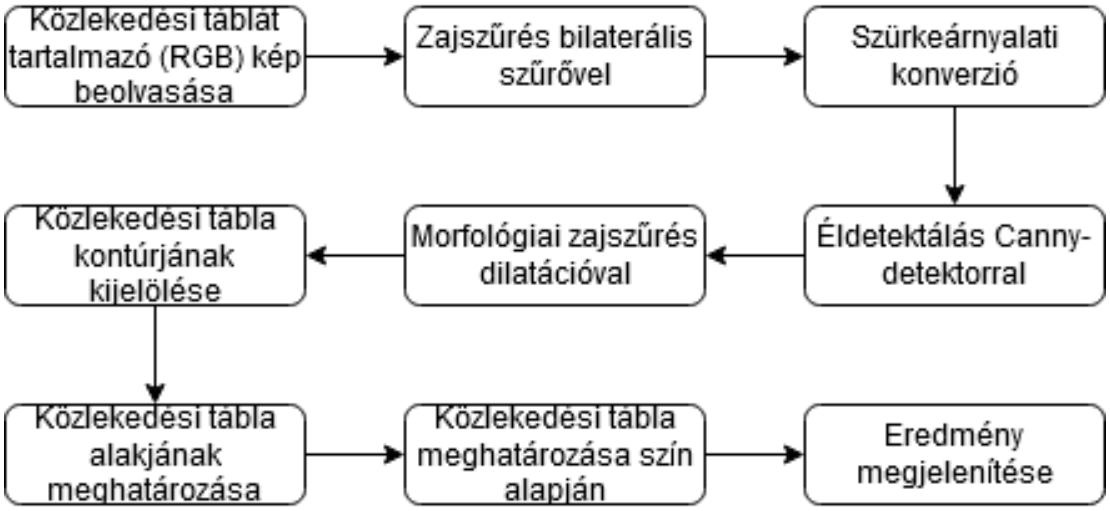
Az éldetektálás Canny-detektorral történik, amely először lineáris konvolúciót hajt végre Gauss-szűrővel, majd gradiens számolás és élerősségszámítást követően eltávolítja a fals élpontokat (nem-maximum vágás). Végül kettős küszöbölést és hiszteréziskűszöbölést végez.

A zajok csökkentése végetti képszűrés során pixelről pixelre haladva a kép egyes pixeleit az azok környezetében lévő pixelek alapján módosítjuk. A Gauss-szűrés esetén a szomszédos pixelek intenzitásának súlyozott átlagát veszik, azaz a közelebbi szomszédok nagyobb súllyal szerepelnek. A bilaterális szűrő egy élvédő, zajcsökkentő simító szűrő, amelynek működése hasonló a Gauss-szűrőhöz, azzal a különbséggel, hogy a súlyokat nemcsak a pixelek euklideszi távolsága határozza meg, hanem radiometriai különbségek is (pl. színintenzitás, mélységtávolság).

A zaj csökkentése végett morfológiai módszerek is alkalmazhatóak, mint például dilatáció. A dilatáció során egy csúsztatott strukturáló elemmel “ütköztettjük” a kép

objektumait, melynek eredménye a kép és a strukturáló elem metszetének nem üres halmaza lesz. [1]

A megvalósítás terve és kivitelezése



A Python nyelven írt program első lépésben a felhasználótól egy képet vár egy fájldialógon keresztül, ahol egy grafikus felületen kiválaszthatja a user a megfelelő formátumú (például .jpeg, .png kiterjesztésű) képet. Ezután a bilateralFilter függvény használatával, azaz bilaterális szűrő segítségével csökkentjük a képen található zajok mennyiségét. Ezt követően a színes (BGR) képet szürkeárnyalatossá alakítjuk. A szürkeárnyalati konverzió a cvtColor függvény segítségével történik meg. A későbbi alakdetektáláshoz a kezdeti képet (*img*) *imgContour* néven másoljuk le. A tapasztalat alapján beállított küszöbértékek szerint történik az éldetektálás. A Canny-éldetektálás a szürkeárnyalatos képen megkeresi az éleket. Az élek detektálását követően a képen további zajszűrésre kerül sor dilatációval.

A program ezt követően megkeresi az éldetektálás után létrejött képen a kontúrokat a findContours függvénnyel, majd a legnagyobb területet bezáró, azaz legjobb kontúr (*best_cnt*) alapján határozzuk meg az alakot. A megtalált, legjobb körvonalat megjelenítjük az *img_contour* nevű képen, amely az eredeti kép (*img*) másolata.

Az alakdetektálás elve a következő: a kontúrt közelítjük egy poligonhoz az approxPolyDP függvény segítségével, majd a visszatérési érték alapján meghatározzuk a tábla alakját. Az epsilon a táblakontúr megengedett eltérését jelenti a sokszög körvonalához

képest. Az approxPolyDP függvény pontok listájával tér vissza, amelynek hosszúságát megállapítva következtethetünk a csúcsok számára, ezzel megállapítva a közlekedési tábla alakját. Ha a csúcsok száma:

- 3, akkor háromszög(*triangle*)
- 4, akkor négyszög (*rectangle*)
- 5, akkor ötszög (*pentagon*)
- 6, akkor hatszög (*hexagon*)
- 8, akkor nyolcszög (*octagon*)
- 10, akkor csillag (*star*)

alakú táblát talált a program. Amennyiben ezen feltételek egyike sem teljesül, a közlekedési tábla alakja kör (*circle*).

A közlekedési tábla körvonala köré a boundingRect függvény egy véletlenszerű színnel jelölt határoló téglalapot rajzol ki. Ezen belül történik a tábla színeinek vizsgálata. A vizsgálat elve a következő: Négy színt vizsgálunk: a kéket, a vöröset, a sárgát és a fehéret. A színek meghatározásának módszere függ a tábla alakjától. Minden esetben megvizsgáljuk, hogy beletartoznak-e valamelyik színkategóriába a téglalap vízszintes tengelyének közepére merőleges szakasz pixelei. Amennyiben nem körről, háromszögről, illetve oktagonról van szó, akkor egy vízszintes szakaszt is vizsgálunk a téglalapon belül, a téglalap függőleges tengelyének középére merőleges vízszintes szakasz képpontjait.

Végezetül a pixelek színei, illetve a tábla alakja alapján kerül meghatározásra a közlekedési tábla.

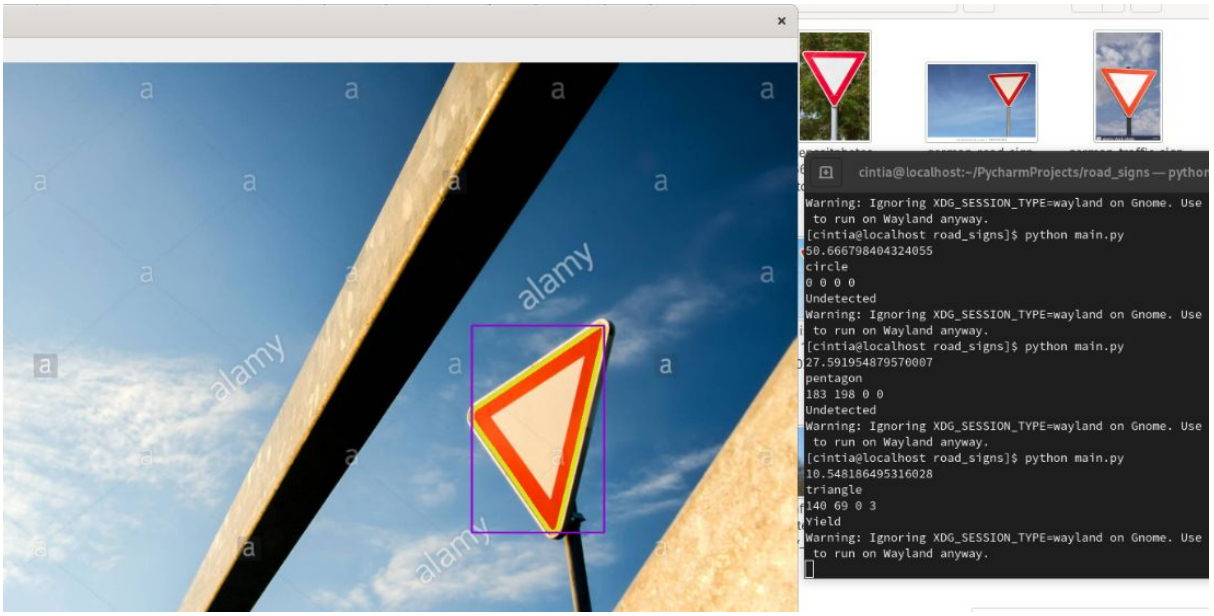
```
road_sign = "Undetected"
if shape == "circle":
    if blue < white < red and white > yellow > 0 and red >= ((white + red + blue + yellow) / 2):
        road_sign = "No Entry"
if shape == "octagon":
    if red > blue and red > yellow and red > white and red >= ((white+red+blue+yellow)/1.5):
        road_sign = "Stop"
if shape == "rectangle":
    if blue < white < yellow and white > red > 0 and yellow >= ((white + red + blue + yellow) / 3):
        road_sign = "Main Road"
    if (yellow < white < blue and yellow < red < blue and
            white > 0 and red > 0 and blue >= ((white+red+blue+yellow)/3)):
        road_sign = "No Through"
if shape == "triangle":
    if blue < red < white and not red <= yellow > 0 and white >= ((white + red + blue + yellow) / 3):
        road_sign = "Yield"
print(road_sign)
```

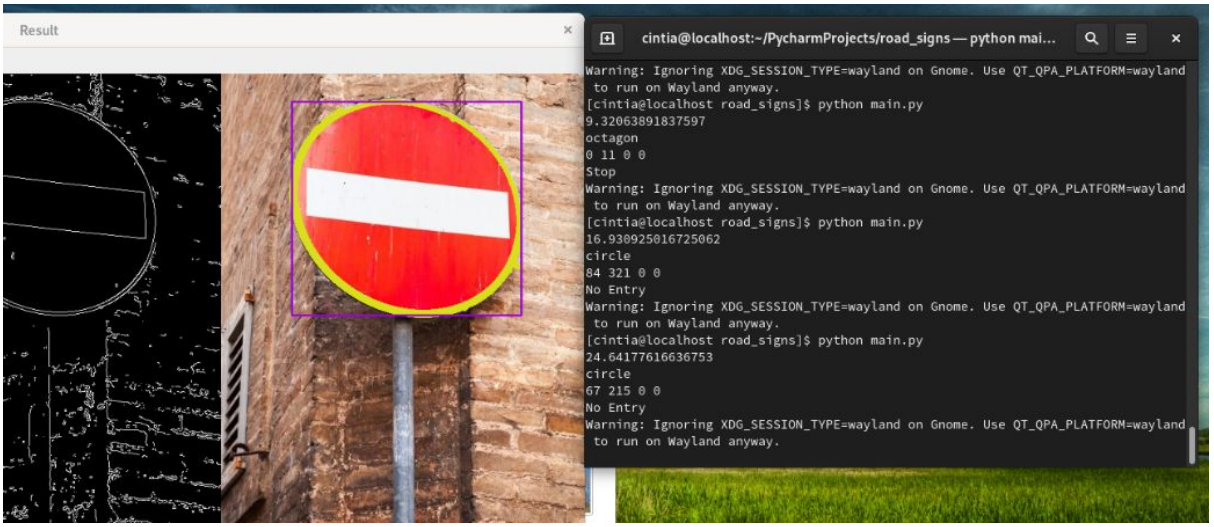
Abban az esetben, ha nem kerül meghatározásra tábla, az *Undetected* felirat jelenik meg.

Végül a stack_images függvénnyel 3 képet összefűzzük: az eredeti, felhasználótól kért inputot; az éldetektálás során keletkezett képet és a táblát kontúrral kijelölő képet, majd együtt jelenítjük meg eredményként. A program addig fut, amíg a ‘q’ billentyű leütésre nem kerül.

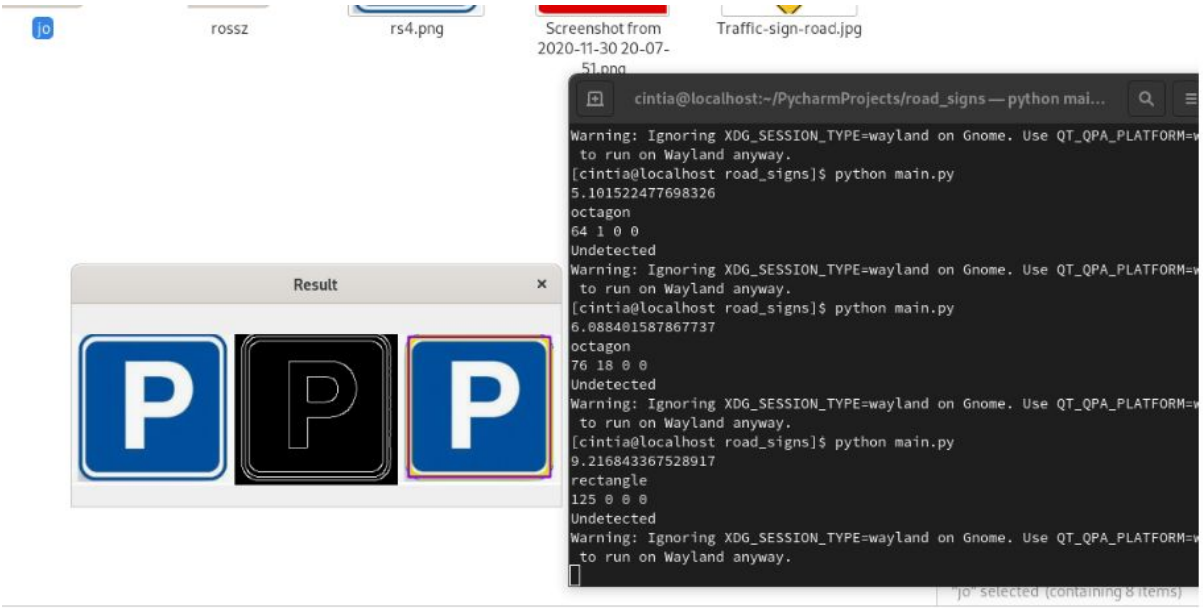
Tesztelés

A közlekedési táblák felismerése azokon a képeken működik jól, amelyeken a közlekedési tábla a legnagyobb objektum, ugyanis a képen található élek közül a legnagyobb kontúrú objektum alapján történik az alak, továbbá a szín meghatározása.

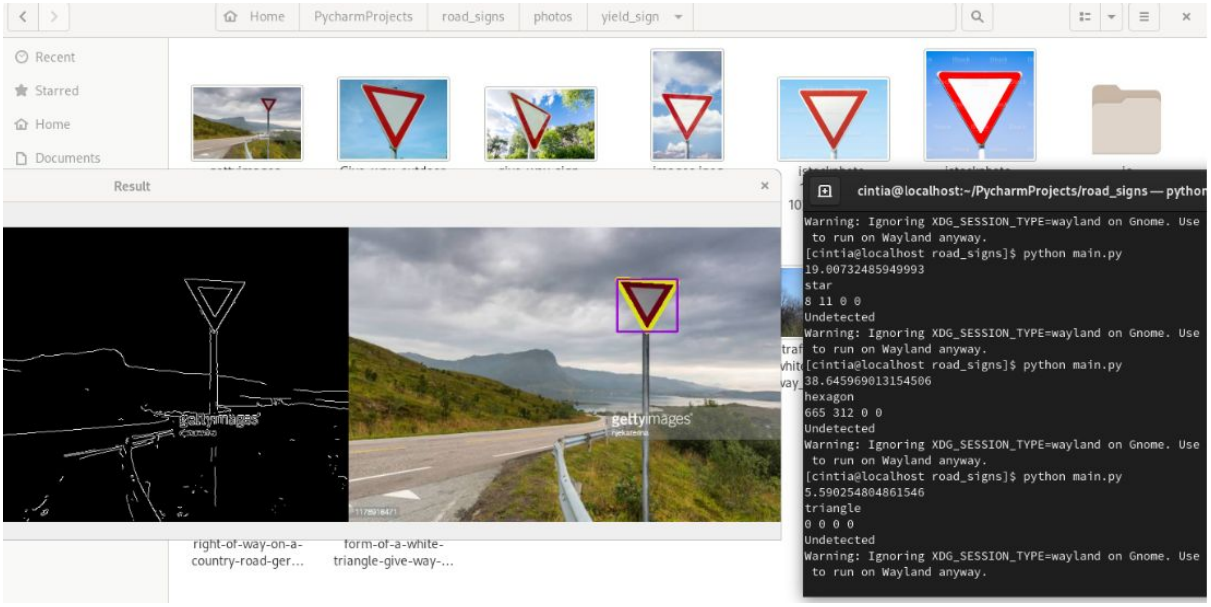




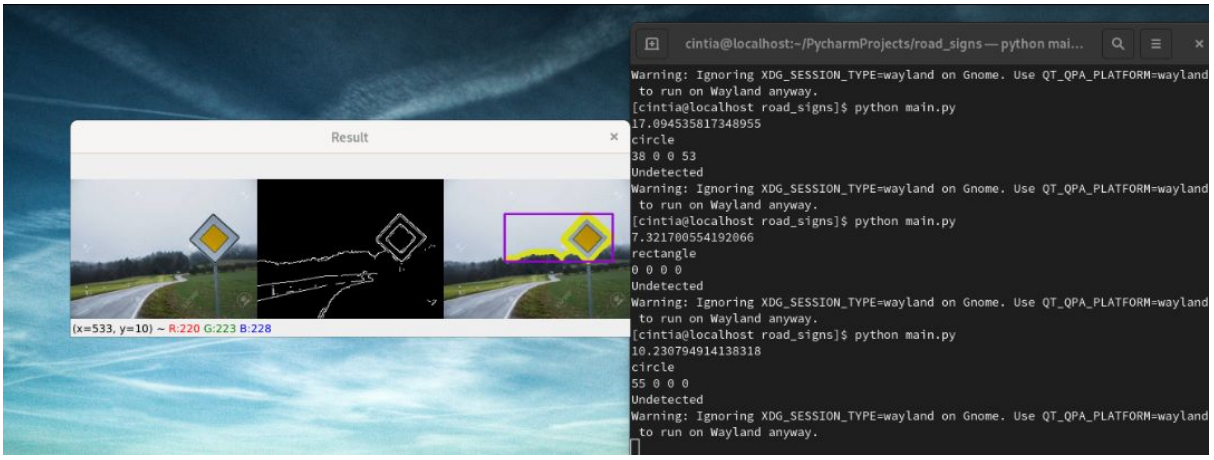
Ha a program nem ismeri fel az előre definiált táblák közül egyet sem, az *Undefined* feliratot jeleníti meg.



Megfigyelésem szerint a táblafelismerés sötét képek esetén nem működik megfelelően a színek helytelen meghatározása miatt.

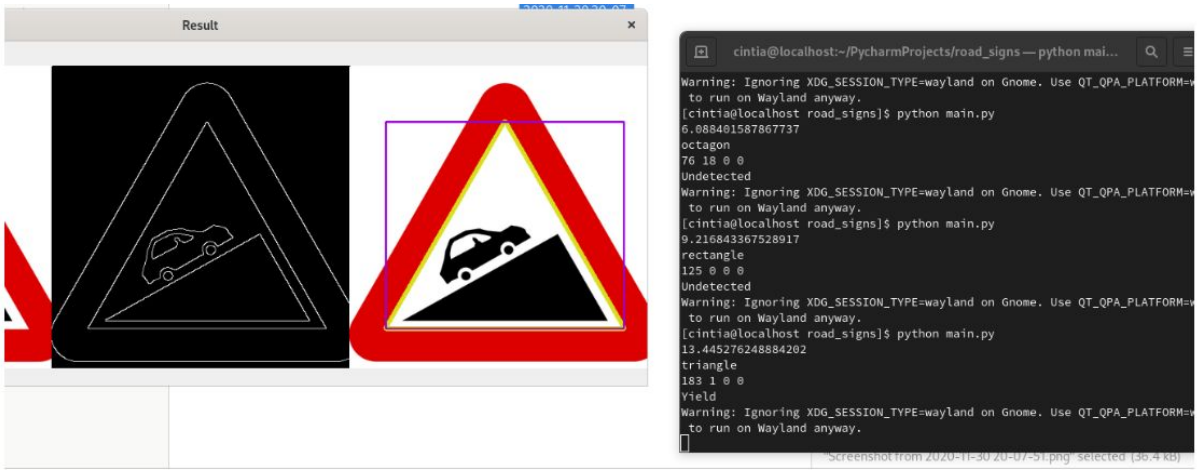


Több esetben problémát jelentett, amikor egy másik tárgy (például belógó ágak, felirat, tartóoszlop) módosította a közlekedési tábla élének alakját, ezzel egy hamis formát jelezve.



Az epsilon, azaz a körvonal poligontól való eltérésének mértéke helyes megválasztása nagyobb tapasztalot, illetve további fejlesztést igényel. Az epsilon megfelelő értékének beállítása függ a kép nagyságától is. A képek méretének normalizálása elképzelhető, hogy megoldást nyújtana erre a problémára. Előfordul továbbá, hogy a Behajtani tilos (*No entry*) és a STOP táblát összetéveszti a program az alak helytelen meghatározása (nyolcszög és kör felcserélése) következtében.

A program egyes esetekben olyan táblát ismer fel, amit nem definiáltam előre. Erre példa a következő kép: Elsőbbségadás kötelező (*Yield*) táblaként határozza meg a Veszélyes emelkedő táblát az alak, illetve a színek arányai alapján.

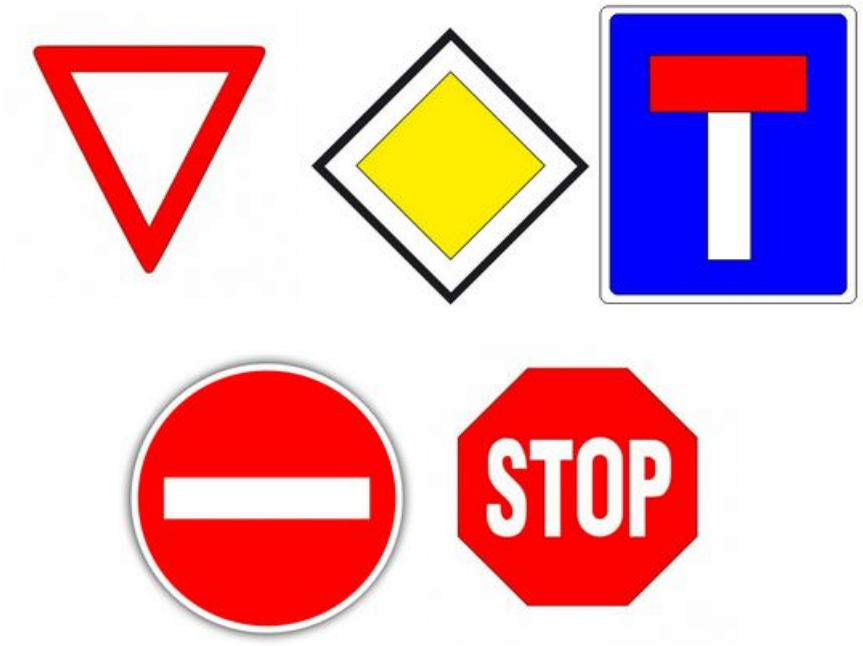


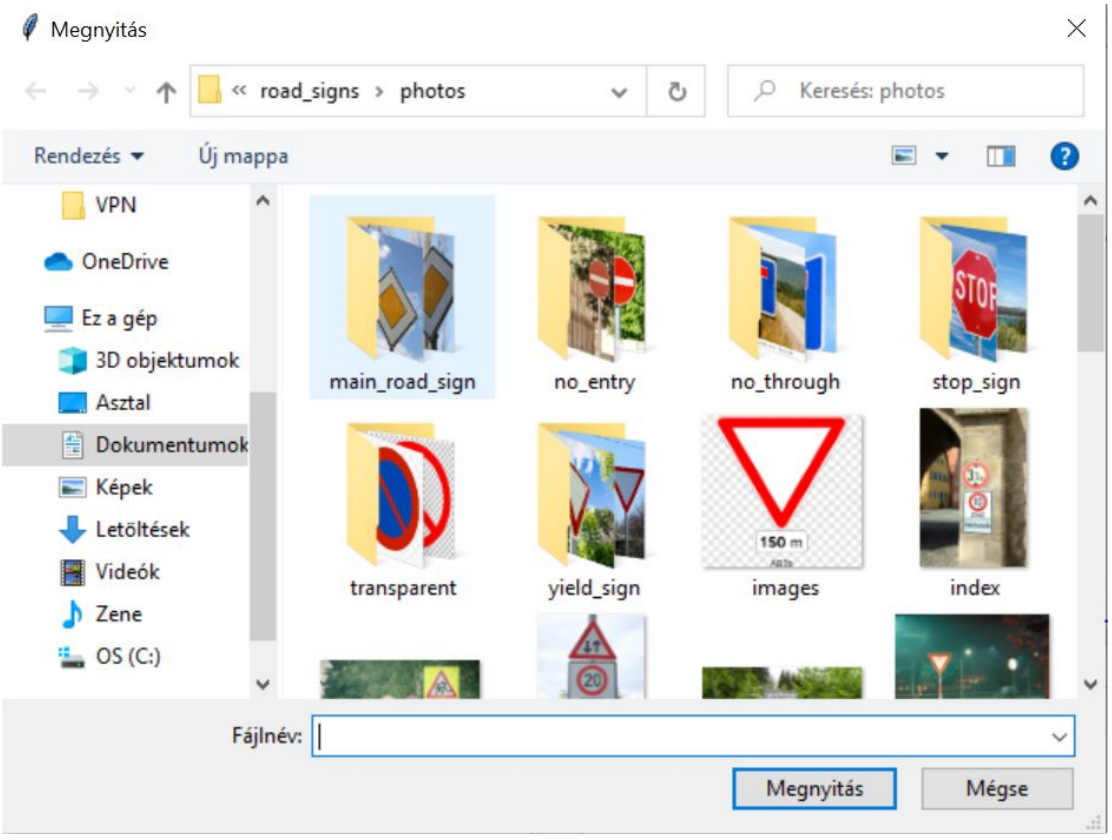
Felhasználói leírás

A program segítségével a következő 5 közlekedési tábla ismerhető fel:

- elsőbbségadás kötelező *(Yield)*
- főútvonal (*Main Road*)
- zsákutca (*No Through*)

- behajtani tilos (*No Entry*)
- STOP tábla (*Stop*)





A program egy fájlválasztó párbeszédpanelen, azaz egy grafikus felületen keresztül képi (például .png, .jpeg kiterjesztésű) bemenetet kér. A kép kiválasztását követően megjelenik egy *Results* címsorú ablak, melyben három kép látható: az eredeti, a felhasználó által kiválasztott, melyen még nem történt módosítás; egy kép, melyen az éleket határoztuk meg, illetve a közlekedési tábla kontúrját tartalmazó kép. Az alábbi esetben látható a program által felismert közlekedési tábla alakja a konzolon: nyolcszög (*octagon*), illetve a neve: STOP (*Stop*).

Abban az esetben, ha a program nem ismeri fel a táblát, az *Undetected* szöveg jelenik meg. A program bezárása a ‘q’ billentyű leütésével történik.

Irodalomjegyzék

[1] Rövid A.–Vámossy Z.–Sergyán Sz. 2014 *A gépi látás és képfeldolgozás párhuzamos modelljei és algoritmusai* Typotex Kiadó

[2] VIDEOPRAKTIKA 2006/4 *Gépi látás I. rész*