Projektbeszámoló
Képfeldolgozás haladóknak MSc gyakorlat

Sajtos Gyula

2015. október 18.

1. Feladatspecifikáció

A projekt témájának megnevezése „Logófelismerés”, azonban ez meglehetősen nagy teret ad az értelmezésnek. Ezért fontosnak tartottuk, hogy pontosan definiáljuk a megoldandó problémát.

A feladatot úgy értelmeztük, hogy adott bemeneti képen a megvalósítandó módszer felismeri az azon látható termék feltételezett márkáját annak logója alapján, és kimenetként megadja annak nevét. Amennyiben több lehetséges termék is található a felvételen, akkor azok közül egyet azonosít. Hogy tovább finomítsuk a specifikációt és egyben csökkentsük az erőforrásigényeket, a tervezett alkalmazás csak alkoholos italok, sörök címkéjén szereplő logó felismerésére készítjük fel. Kibővítettük a specifikációt felhasználói beavatkozással a logó helyének kijelölésében.

2. Fejlesztési folyamat

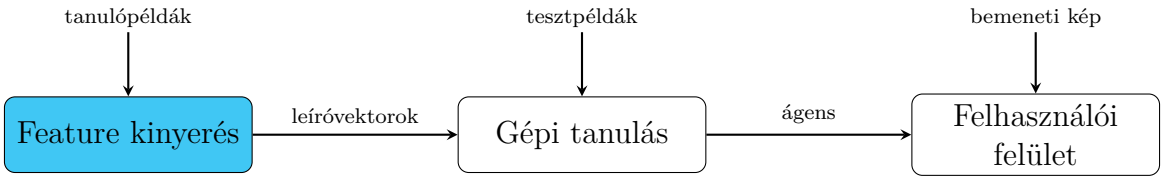
2.1. Környezet

Az alkalmazható módszerek teszteléséhez és prototipizáláshoz a Matlab csomagot választottuk masszív képfeldolgozási eszköztára miatt. A megbeszélések során felvetődött az OpenCV keretrendszer használata. A csapat közös döntése értelmében ennek alkalmazását csak egy jól teljesítő prototípus elkészítése utánra halasztottuk. A fejlesztési folyamat során előálló dokumentumok és forráskódok verziókezelésére a webes GitHub szolgáltatást jelöltük ki.

2.2. Részfeladatok

A probléma jól meghatározható részekre bontható, ennek megfelelően lettek kiosztva a csapat tagjainak – a részproblémák körülhatárolásában segédkeztem. Az 1. ábrán láthatóak az egyes részproblémák. A megjelölt komponens az én felügyeletem alá esik, a gépi tanulási feladat *Kiss Marcell*, míg a GUI elkészítése *Ábrahám Gábor* hatáskörébe tartozik. Az én általam adott kimenetek a gépi tanulási stádium számára megadják a képi adatbázis elemeinek leírását. Ebből készül egy modell, ami az osztályozást fogja végrehajtani. Végül a felhasználói felület fogja megadni a logóra vonatkozó becslést, vagyis

milyen márka címkéje látható a képen. Ehhez nyilván szükség van magára a döntéshozó ágensre, valamint a felhasználó által kijelölt régió (ROI) jellemzésére.



1. ábra. A logófelismerési feladat komponensei

2.3. Módszer

A következőkben röviden kifejtem a részfeladatomhoz tartozó problémákat és a megközelítésemet. Jelenleg a SIFT [1] és SURF [2] feature leírók használatával kapcsolatos tesztelések folynak. Ezen módszerek illenek a probléma természetéhez, mivel skála- és forgásinvariánsak, sőt bizonyos mértékig ellenállók az affin transzformációkkal, a különböző nézőpontokból származó torzulásokkal szemben is. Problémát jelentenek a többcsatornás képek, mivel színes képpel számoltunk a bemenetet illetően, azonban az említett detektorok csak szürkeárnyalatos képre specifikáltak. Erre két megoldás van: szürkeárnyalatra alakítás vagy más színtérbe lépés – mindkettőt bevetettem. Ha ránézünk egy márka címkéjére, szinte biztos, hogy az azonosításában szerepe van az egyes színeknek. Ez nem csak az emberi látás számára jelent támpontot, hanem számos osztályozónál, a jellemzők elválasztásánál. Az RGB $\rightarrow$ HSV konverziót választottam, mivel az RGB reprezentáció nem tükrözi jól a homogén színű területeket. Jól demonstrálja az 2. ábra a hue csatorna szerepét a SURF detektor esetén.

Azonban még így is akadt probléma: a három komponenst hogyan ábrázoljuk? Eddig két kezdetleges módszer került a kódba. Egyiknél a hue, saturation és value csatornákon külön-külön detektált pontokból kinyert vektorok egymás után kerülnek a tanulóadatbázisba. A másik esetben egy vektor tartalmazza a három csatornán kinyert információkat, tehát – kiindulva a Lowe által használt 64-komponensű leírókból – 128-hosszú jellemzőleírók jönnek létre. A korábbi megnehezítheti az osztályozó feladatát, ha például több hue és saturation vektor eltérő osztályokból egymáshoz közel kerül, így torzítva a felosztást. Utóbbinál ez nem eshet meg, de még így is finomítást kíván. Jelenleg dolgozom egy olyan verzión, amiben valamely csatorna (erre a hue-t tartom jó választásnak) detektált pontjaiban hajtódik végre a feature kinyerés, így egy vektor csakis egy ponthoz fog tartozni, viszont a másik két csatornáról is lesz gradiens-információ. A szakirodalom szerint objektum-felismerésben jól teljesít a SIFT leíró egy változata, mely kissé eltérő stratégiát használ a pontok kiválasztásánál (sűrűbb léptékkel nyeri ki a jellemzőket, így kifinomultabb leírását kapjuk a tárgynak). Ezt is szeretném belevenni a megvalósításba, ha elegendő idő áll rendelkezésre a SIFT beépítése után.

Ezen módszerek robusztusak, ezáltal implementációjuk (numerikus és tesztelési szempontból) sok kihívást jelent. Már egynek a megvalósítása is felemésztené a rendelkezésre álló időt, így ezen esetekben külső könyvtárakhoz fordultam. Szerencsére elérhetőek implementációk: a SURF megtalálható a Matlab képfeldolgozó csomagjában, a SIFT-hez pedig szintén készült megvalósítás[1].

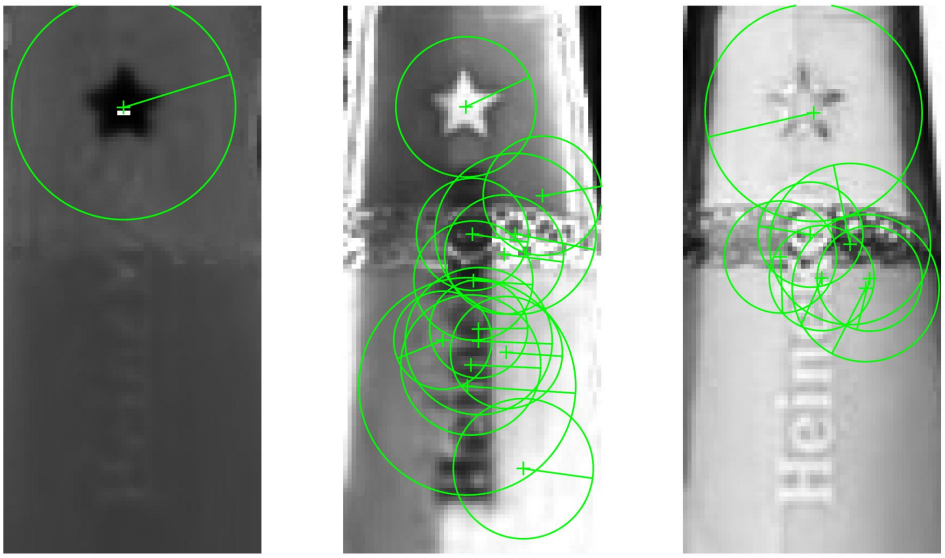
[1] http://www.vlfeat.org/

Hogy olajozottan menjen a fejlesztési folyamat, egy függvényben egyesítettem a különböző feature kinyerési módszereket, így az utána következő stádiumban csak a paramétert kell változtatni kísérletezés közben. A `extractFeatureVecs()` függvény, mint neve is mutatja, a jellemzővektorok kinyerésére készült. Egy mappát és a leíró nevét várja paraméterül. Meghívásakor bejárja a bemeneti `imagePath` mappát, majd – feltételezve, hogy az egyes tanulópéldányok címkéjüknek megfelelő nevű mappában vannak – kigyűjti a címkeneveket és azonosítókat rendel hozzájuk. Ezzel egy időben minden képen végrehajtja a feature detektálást és kinyerést. A kinyert vektorokat egy `allFeatures` nevű mátrixhoz csatolja, ez az egyik kimenete a függvénynek. Hogy lehetővé tegye a betanulást a következő szakasznak, a kimeneti `allLabels` oszlopvektorba szimultán beleteszi a példány osztályának azonosítóját is. Nyilvánvaló, hogy az `allFeatures` és az `allLabels` ugyanannyi sorból áll. Ezek birtokában el lehet végezni a betanítást és a modell kiértékelését is. Szükséges, hogy jól modulálhassuk a betanulást – az én részfeladatomban annyit tehető ez ügyben, hogy a jellemzővektor komponenseinek számát (`featureVecLength`), továbbá a kinyert pontok számát (`pointsPerImage`) korlátoztam egy opcionálisan megadható paraméterrel. Így szabályozható, hogy hány pont vegyen részt az ágens tanulásában képenként.

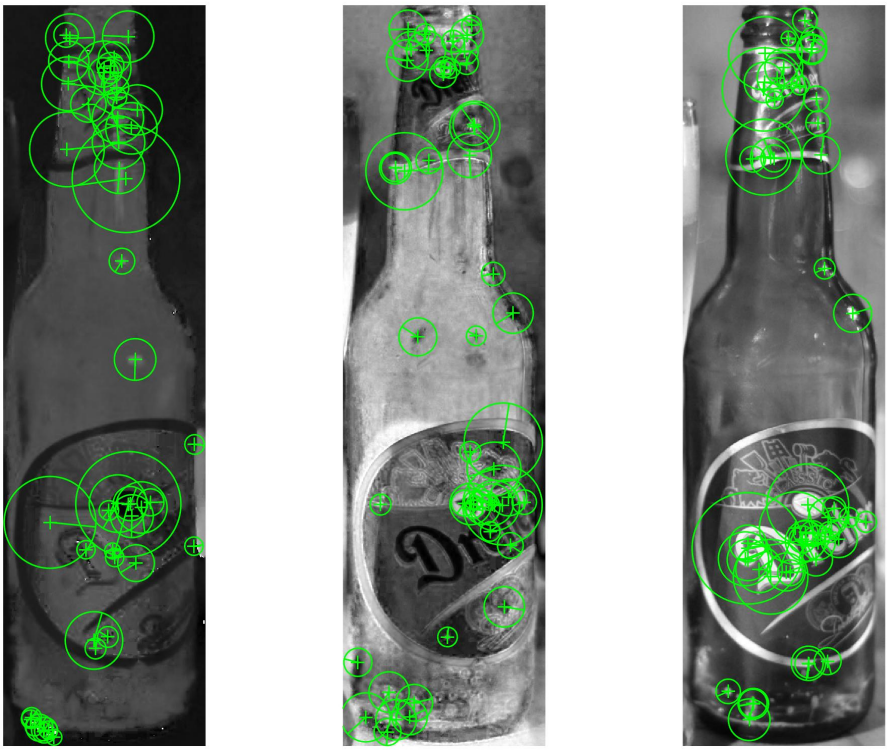
3. Eredmények

Az alábbi képeken látható a kísérletezés során kapott kimenet egy példára. A SURF detektorral észlelt pontok felső korlátja 50 volt. Jól látszik ezen a példán, hogy sikerül elkapnia a logó emberi szem számára is jellemző pontjait.

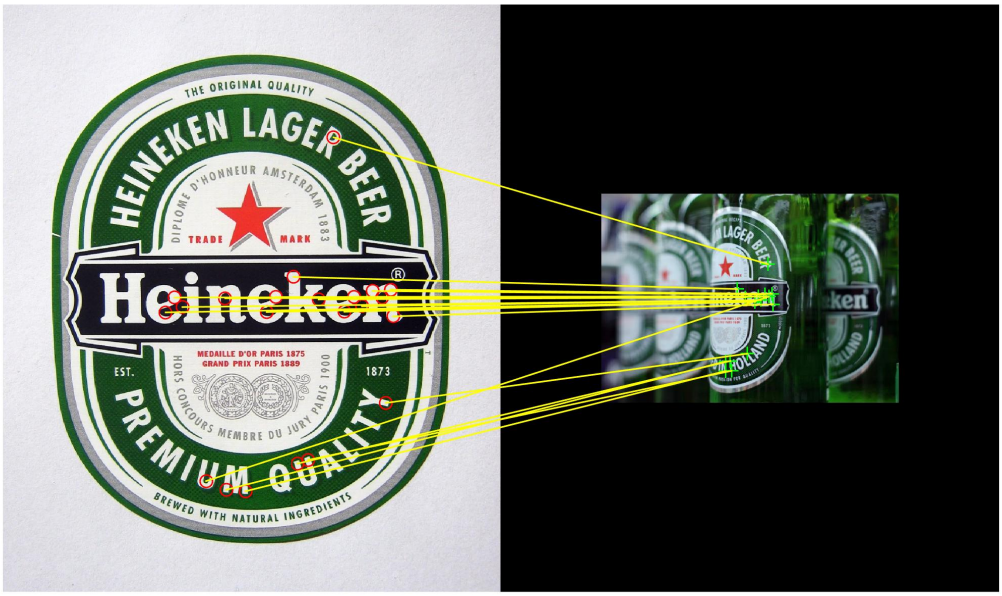
A SURF számunkra nagy előnyét a 4. számú ábra érzékelteti. A baloldalon egy nagyjából steril kép, a jobboldalon pedig egy valós helyzetbeli példa. Ezen kép tanuló algoritmus nélkül, pusztán a két kép domináns feature pontjai összepárosításával készült. Ebben az esetben 100 volt a pontok maximális száma.



2. ábra. Hue, saturation és value csatornán detektált SURF pontok



3. ábra. Hue, saturation és value csatornán detektált SURF pontok



4. ábra. SURF pontok párosítása

Hivatkozások

[1] Lowe, David G. "Distinctive image features from scale-invariant keypoints." *International journal of computer vision* 60.2 (2004): 91-110.

[2] Bay, Herbert, Tinne Tuytelaars, and Luc Van Gool. "Surf: Speeded up robust features." *Computer vision–ECCV 2006*. Springer Berlin Heidelberg, 2006. 404-417.