Dátum eltüntetése fotókról

Készítették:
Dobó László
Munkácsy Gergely
Sebők Ágota

Kontakt:
Németh Gábor

Weboldal:
http://festo.github.com/Diana/

Készült:
2012. December 9.

A feladat leírása

A modern fényképezőgépek nagy része képes dátum címke elhelyezésére a képeken, amit utólagosan már nem lehet kikapcsolni. Ezeknek az eltüntetése kézzel nagyon körülményes ezért a feladatunk az, hogy keressünk egy megoldást ennek a feladatnak az automatizálására.

Programozási környezetnek először Java-t választottunk. Ezen belül ImageJ plugint szerettünk volna készíteni de még a munka elején áttértünk MATLAB-ra, mert a beépített képfeldolgozó algoritmusok kezelték a színes képeket is az eredeti nyelvvel ellentétben.

Megvalósítás, problémák, eredmények

A megvalósítás folyamatát két jól elkülöníthető részre lehet bontani az alapján, hogy hogyan dolgozzuk fel a képeket. Első lépésként a dátumot kell detektálnunk, majd az így kapott maszk alapján le kell retusálnunk a képről a megadott területet.

A maszk előállítása

Követelmények, kikötések

A feladatot megnehezíti az, hogy a különböző fényképezőgépek különböző méretben, színben és más pozícióban helyezik el a dátumot a fényképeken. Erősen befolyásolja az eredményt az, ha a kép túl világos vagy túl sötét vagy eléggé textúrázott az időbélyegző mögötti terület. Számos paraméter befolyásolja az időbélyegző detektálását, de vannak közöttük olyan tulajdonságok, melyekre tudunk megszorítást adni és nem rontanak az algoritmus használhatóságán. Ezek a feltett és általunk lerögzített paraméterek a későbbiek folyamán bármikor kibővíthetők a teljes, legbővebb paraméter halmazra.

Feltesszük azt, hogy a dátum a kép jobb alsó sarkában van. Ha álló képről beszélünk akkor a kép bal oldalán, függőlegesen, pont úgy, hogy egy transzformációval az előző állapotba lehessen hozni a képet.

A detektálás folyamata nem függ a használt betűtípustól és a dátum méretétől, valamint a felirat színétől, de azt feltehetjük, hogy a dátum bélyegzőnél használt szín intenzív, homogén, például piros vagy sárga. Mi az előbbi két színre teszteltük az algoritmust.

Az algoritmus

Ezek alapján első lépésként ha a kép álló formában van akkor elforgatjuk, majd kivágjuk a kép alsó 10%-át és a szélességét is jobbról lekorlátozzuk az eredeti 40%-ára. A továbbiakban az így átméretezett képpel dolgozunk.

Mivel a legtöbb esetben a dátumok erősen elhatárolódnak a háttértől ezért morfológiai operátorokkal még jobban kiemeltük a szöveg éleit. Egy morfológiai nyitást hajtunk végre a képünkön, majd egy zárást és ezt kivonjuk az eredeti képből. Ez a módszer a sötétebb képeken kiemelte a keresett szöveget, de a világos képeknél nem adott jó eredményt.

Dátum = Kép - Zárás(Nyitás(Kép))

A két morfológia operátor lefuttatása és a különbségképzés után egy u.n. Saliency map algoritmust futtatunk a vizsgált területen a kiugró, intenzív pixelek megtalálására. A megtalált képpontok között lesz nagy valószínűség szerint a dátum bélyegző is. Az algoritmus megfelelő paraméterezés után olyan szürkeárnyalatos területet ad vissza, amit mi keresünk. A Saliency map algoritmus futtatása után a kapott eredményt binarizáljuk, 50%-os köszöböléssel, így kapjuk meg a maszkot. Viszont sok esetben az eredmények nem kielégítők, ezért valamilyen módon finomítani kell a detektálási folyamatot.

Problémák, a módszer finomítása

Megpróbáltuk a kép rétegeit külön külön is vizsgálni, de arra az eredményre jutottunk, hogy hiába végezzük el a morfológiai műveleteket az RGB kép színcsatornáin egyesével, az eredmény nem sokban változik, ugyanis a dátum szín információja általában egy adott csatornán szerepel az RGB három csatornája közül. (Intenzitás) A piros dátumról a legtöbb esetben az R csatorna tartalmazza a legtöbb információt. A másik két csatornán található információ mennyisége elhanyagolható. Ezért nem ad jobb eredményt, ha csatornákra bontjuk a képet. Hasonló a helyzet ha egyes rétegeket figyelmen kívül hagyunk vagy párosával vizsgáljuk őket.

Az nem jó megoldás, ha ilyenkor csak azt az egy csatornát vizsgáljuk, amelyiket a legtöbb szín információ van a dátumból, ugyanis a kép globális tulajdonságai közül sokat elvesztünk, mint például a dátum mögött elhelyezkedő textúra, ami létfontosságú a detektálás szempontjából.

RGB használata esetén az algoritmus fekete, homogén hátterű képeknél nagyon pontos találtatot adott, az intenzív kontrasztok miatt, de világos, erősen textúrázott hátterű képek esetében nem. Ezért próbáltuk ki a HSV színteret. (H: színányalat, S: telítettség, V: érték)

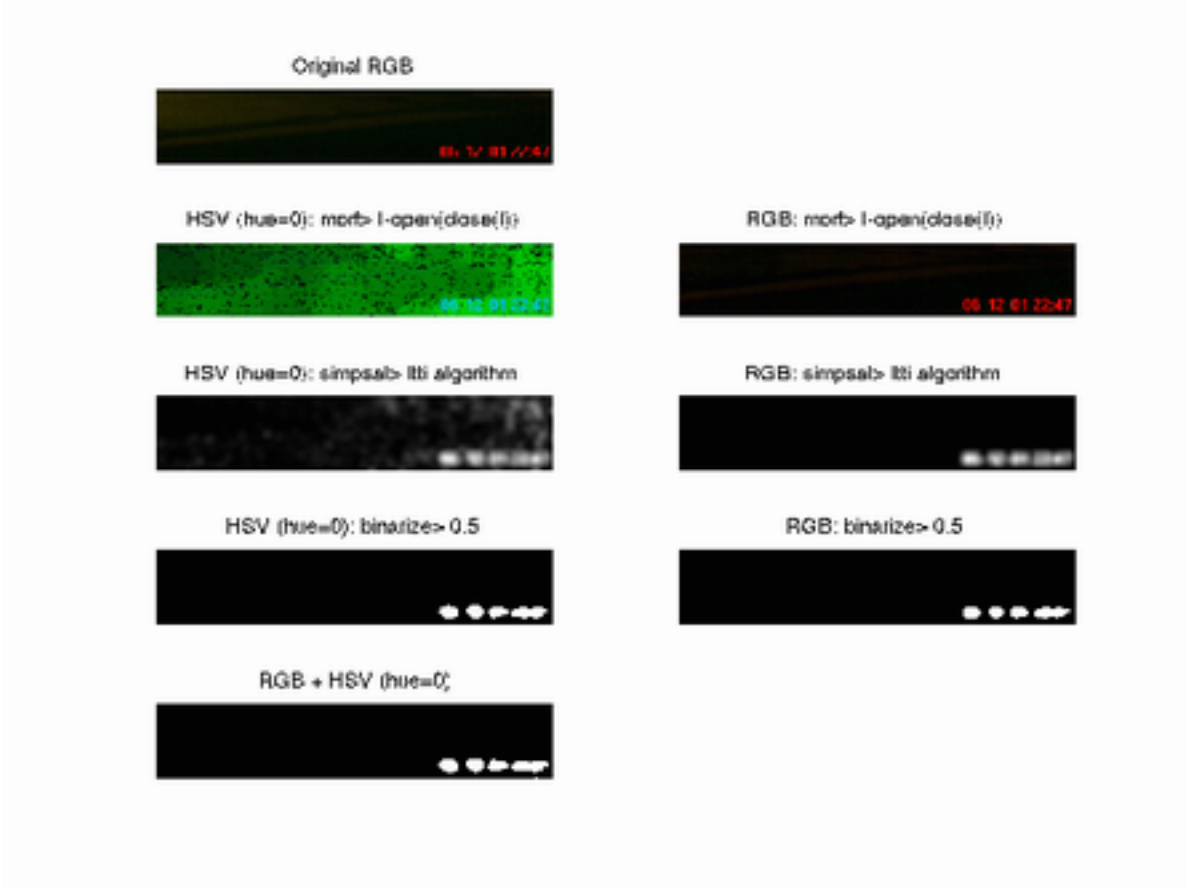
Ha HSV csatornákkal foglalkoztunk, akkor jobb eredményeket értünk el számos esetben. Próbálgatás után rájöttünk, hogy a színárnyalat információkat elhagyhatjuk a színinformációk közül (A HSV első csatornáját), így kapjuk a legjobb eredményt. RGB-n és HSV-n külön külön futtattuk az algoritmust, ebből kaptunk egy-egy maszkot, majd a két maszknak képeztük az unióját. A továbbiakban azt kell vizsgálnunk, hogyan kapunk jobb eredményt.

Eredmények

Az eredményeink változatosak voltak, amit a következő táblázat mutat be:

Felirat	Tesztek	RGB	HSV	HSV + RGB*	Rossz
Sárga (Kézi)	40	0	0	27 (68%)	13 (32%)
Piros (Gépi)	26	3 (11%)	6 (23%)	17 (66%)	0
Összesen	**67**	**3 (4%)**	**6 (9%)**	**44 (68%)**	**13 (19%)**

A tesztelés utáni következtetés levonás szubjektív, szabad szemmel történt az eredmények helyesség vizsgálata. Az általunk legjobbnak ítélt maszk kapta meg a pontot. RGB jelöli az RGB színtér használatakor keletkezett maszkot (a HSV-re hasonló módon), a HSV + RGB pedig azokat az eseteket, amikor vagy a két maszk uniója bizonyult jobbnak. (Ide számít az is amikor a két maszk hasonló eredményt adott, ilyenkor is el tudjuk fogadni az unió által létrehozott eredményt. A Rossz oszlop jelenti azt, hogy téves eredményt adott vissza mindhárom maszk.



Az esetek túlnyomó többségében vagy a két maszk uniója bizonyult a jobbnak, ezért a továbbiakban ezen maszkot vizsgáljuk tovább, az így kapott eredményeket már az esetek túlnyomó részében el tudjuk fogadni.

A létrejött maszk sokszor tartalmaz szemmel megállapíthatóan nem a dátumot lefedő területet (ahhoz nem kapcsolódó képpontokat). Ezért az eredményt tudjuk finomítani azzal, hogy vesszük a legnagyobb, összefüggő területet és annak a legkisebb befoglaló téglalapját. A területről tudjuk, hogy téglalap alakú és az oldalak aránya körülbelül 1: [10, 20], valamint a kép jobb alsó oldalához közel, fektetve helyezkedik el. Az ezen kívül eső pontokat elhagyjuk.

Inpainting

Az előző maszkot megkapva, a feladat az, hogy a detektált területet, a dátum képpontjait kitöltsük, kiretusáljuk. Ehhez a határos képpontok információit, a háttér textúráját használjuk fel.

Az inpaintig technikát alapvetően egy kép hiányzó vagy sérült részeinek helyreállítására használják: festmények restaurálására csakúgy mind digitális képek javítására. Jelen esetben a dátum jelenti a sérülést.

A Matlab-ban számos képfeldolgozó eszköz megtalálható, köztük a roifill, amely kisebb területek befoltozására kiválóan alkalmas.
Az eszköz működésének lényege, hogy a paraméterként kapott, eredeti képpel azonos méretű bináris maszk nemnulla pixelei alapján interpoláció segítségével kiszámolja a maszk által fedett területen lévő pixelek értékét, kívülről befelé haladva, a Laplace formula szerint.
A problémát az okozta, hogy a határon lévő pixelek alapján számolt, így a határpixelek változatlanok maradtak és mivel a dátum detektálása igen nagy hatékonysággal szűrte ki a dátumot, ezért sokszor az élénk színértékek nagyban befolyásolták az új értékeket. Ezért a maszk méretét meg kellett növelni pár pixellel és így már elfogatható eredmény született.
Ezt végre kellett hajtani az eredeti RGB kép színcsatornáin egyesével, majd újraegyesítéssel létrejött az immár retusált végeredmény.

Felhasznált irodalom

- Photo time-stamp detection and recognition - Xiangrong Chen and Hong-jiang Zhang
- Bayesian Approach to Photo Time-Stamp Recognition - Asif Shahab, Faisal Shafait, Andreas Dengel
- MATLAB Saliency - http://www.klab.caltech.edu/~harel/share/gbvs.php
- http://www.mathworks.com/help/images/ref/roifill.html