konferencia szemle

HTE MediaNet 2015

Diákszekció

2015. október

konferencia
szemle

TARTALOM

HTE MEDIANET2015 KONFERENCIA | **DIÁKSZEKCIÓ**

Eredmények a többutas hálózati kommunikációs technológiák területén

Fejes Ferenc, Katona Róbert, Püsök Levente

***Abstract*—A többutas hálózati technológiák az elmúlt néhány évben nagyon aktív kutatási területté váltak. A jelenlegi hálózati technológiák nem minden esetben képesek kielégíteni a gyorsan növekvő igényeket. Az Internet alapvető működése örökölte azt a filozófiát, ami a több mint 30 éve, a hálózati és szállítási protokollok tervezése idején reális volt: a hálózati csomópontok egyetlen interfészt használva, egyetlen útvonalon kommunikálnak egymással. A mai környezetben ez már nem feltétlenül így van, rengeteg eszköz több hálózati interfésszel rendelkezik, több kijárata van az Internetre, ezzel megteremtve a lehetőséget, hogy akár egy kommunikációs viszonyon belül több útvonalon keresztül történjen meg az információcsere. Ebben a cikkben röviden áttekintjük napjaink legfontosabb adatkapcsolati és transzport rétegben működő többutas kommunikációt támogató megoldásait. Továbbá bemutatjuk a Debreceni Egyetem Informatikai Karán kifejlesztett MPT-GRE - Multipath Communication Library-t, ami egy hálózati rétegben működő többutas kommunikációt támogató eszköz. Korábbi publikációk eredményei azt mutatják, hogy az MPT alkalmas különböző hálózati interfészeken megvalósított kommunikációs útvonalak kapacitásának hatékony aggregációjára. A szoftver emellett alkalmas a több út redundáns módú használatára is. Ebben a cikkben az MPT ezen funkcionalitását vizsgáljuk: egy notebook Wi-Fi és 3G mobilinternet kapcsolatai között váltunk videóstream átvitel közben. Az eredmények azt mutatják, hogy a Wi-Fi interfész tervezett lekapcsolása esetén a videóstream továbbítás csomagvesztés mentesen kerül át a 3G interfészen felépített útvonalra, biztosítva ezzel folyamatos és akadásmentes kép és hang lejátszást a kliens oldalon.**

***Keywords*—multipath communication, IEEE 1905, MPTCP, GRE in UDP tunnel, vertical handover.**

I. Bevezető

Évről évre növekszik a végfelhasználók által generált internetes adatforgalom. A növekedés hátterében nagyrészt a gyors hálózati kapcsolattal (LTE) felszerelt mobiltelefonok állnak. Az LTE előfizetések száma 2013-ról 2014-re több mint duplájára nőtt, a mobil adatforgalom ugyan ezen időszakban a havi átlagos 1.5 exabyte-ról 2.5-re emelkedett és egyes előrejelzések szerint 2019-re ez a szám tovább növekszik majd egészen 24.3 exabyte-ra [1]. Az otthonokban is előállt egy olyan heterogén hálózati környezet, amelyet alkotó technológiák nem lettek felkészítve az új felhasználói igényekre (például az Ultra HD felbontású 3D-s videólejátszás, lakáson belüli tartalom streamelés, stb.). Ahhoz, hogy a megnövekedett igények hatékonyan kiszolgálhatók legyenek, az infrastruktúra fejlesztése mellett a már meglévőnek a hatékonyabb kihasználása is egyre fontosabbá válik. A hálózatokban jelenleg is meglévő, de ki nem használt potenciál kiaknázását is célul tűzik ki az ún. többutas (multipath) hálózati megoldások. A végpontokon megjelenő több hálózati interfész integrációja koncepcionális lehetőséget nyit ezek együttes, párhuzamos használatára. A mobiltelefonok nagy része beépítve tartalmazza a 3G, LTE modemet és Wi-Fi kapcsolatra is alkalmas; ugyanez igaz egyes táblagépekre is. A notebookok hosszú ideje rendelkeznek Wi-Fi és RJ-45-ös vezetékes Ethernet kapcsolódási lehetőséggel, illetve USB 3G/LTE modem segítségével is kapcsolódhatnak a kommunikációs világhoz. Ezek a technológiák lehetővé teszik, hogy az internetre egyszerre több kijárati ponton is csatlakozhassunk és egy távoli végpont több útvonalon is elérhető legyen. Az ilyen hálózati környezet több előnnyel is bír: felhasználható arra, hogy egy esetleges útvonal sérülés esetén egy másik útvonalon gyorsan tudjuk folytatni a kommunikációt, vagy több útvonalat szimultán használva azok sebességét aggregálni tudjuk, gyorsabb átviteli sebességet elérve, mint az egyes, különálló utak esetén. Az alkalmazási megoldás függ attól, hogy melyik OSI rétegben értelmezzük ezt a funkcionalitást. Adatkapcsolati rétegben a több út fogalma az otthonokban napjainkra előállt heterogén hálózati környezethez oly módon köthető, hogy az eszközök tipikusan több médiumon is képesek adatátvitelt megvalósítani. Ezt próbálja egységesíteni az IEEE 1905.1a szabvány [2], ami a hálózati réteg (layer 3) számára transzparens absztrakciós réteget definiál a heterogén adatkapcsolati alrétegek fölé. A megoldási mechanizmus vázát röviden áttekintjük a II. fejezetben. Transzport rétegben működő megoldás az MPTCP [3], amely képes a több útvonal hálózati kapacitásának az összegzésére és a közöttük történő gyors váltásra, úgy, hogy több TCP alfolyamot (TCP subflow) is felépít és ezek között hatékonyan osztja szét a továbbítandó adatcsomagokat (ld. III. fejezet). Koncepcionálisan hasonló célra, de a hálózati rétegben nyújt megoldást a Debreceni Egyetem Informatikai Karán kifejlesztett MPT-GRE - Multipath Communication Library (továbbiakban MPT) [4]. Ez a megoldás tetszőleges, akár heterogén verziójú (IPv4, IPv6) hálózati útvonalak fölött valósít meg GRE in UDP tunneling [5] eljárással adatátvitelt. A IV. fejezetben bemutatjuk ennek a technológiának a működési vázát és a többutas kommunikációs méréseink eredményeit. A munka folytatásának további lépéseire az V. fejezetben tekintünk előre.

II. IEEE 1905 absztrakciós réteg

A manapság kapható olcsó vezeték nélküli router-ek tipikusan fel vannak szerelve vezetékes Ethernet switch-el, így egy ilyen eszközt beüzemelve már két médium is rendelkezésre áll adattovábbításra, két különböző adatkapcsolati protokollt

Fejes F., Katona R. és Püsök L. a Debreceni Egyetem Informatikai Karának hallgatói,
e-mail: fejes@opmbx.org, robert.k@opmbx.org, pusok.levente@opmbx.org
Érkezett: 2015. 09. 16.

használva (vezeték nélküli IEEE 802.11 illetve a vezetékes IEEE 802.3 Ethernet valamelyik verziója). További lehetőség egy HomePlug [6] kompatibilis (IEEE 1901 Broadband Powerline Standard szabványt [7] támogató) eszköz beszerzése, ezzel már a ház villanyáram hálózatát is használhatjuk adattovábbításra. A powerline kommunikációs szabványt támogató eszközök alkalmazásával a csavart érpár hálózat kiépítési költségei megspórolhatók. Léteznek olyan AP-k (Access Point) is, amelyek a vezetékes Ethernethez nem csak egyszerű vezeték nélküli hozzáférést biztosítanak, hanem a lakás villamos hálózatán keresztül is megosztják azt, így a teljes lakás vezeték nélküli lefedettséghez elég még néhány további ilyen, villamos hálózatba csatlakoztatott AP és ezek közül elegendő egyikbe becsatlakoztatni a vezetékes Ethernetet. Elterjedt még a háztartásokban a koaxiális kábelezés a kábeltévé szolgáltatásoknak köszönhetően, valamint a televízió antennák elterjedt csatlakoztató médiumaként. Ennek a fizikai közegnek a hatékony kihasználására hozta létre a Multimedia over Coax Alliance (MoCA) [8] saját szabványát, ami verziótól függően eltérő (MoCA 1.1-nél 175 Mbps, MoCA 2.0-nál már 400 vagy 800 Mbps), de gyors és megbízható (MoCA 2.0 szabvány 100 millió adatcsomagonként egyetlen hiba [9]). A szabvány célja a hatékony nagy felbontású otthoni média átvitel, legyen az TV adás, fényképek, videók, zenék átvitele mindezt nagyon alacsony késleltetéssel, hogy alkalmas legyen játékok nappaliba streamelésére is. A felsorolt technológiák jelen vannak a lakásokban, egy erősen heterogén hálózati környezetet alkotva. Ahhoz, hogy a médiumok kapacitásait kihasználjuk, rendelkeznünk kell a hozzájuk szükséges hardveres és szoftveres interfészekkel, ami sok esetben nem megvalósítható, például egy notebook vagy mobiltelefon hálózati interfészei utólagosan nem bővíthetők. A végfelhasználók számára további nehézségeket okoz, hogy minden technológia eltérő konfigurálási módszerrel rendelkezik. További probléma az is, hogy bár a technológiák együttes lakásbeli lefedettsége nagyon magas, külön-külön viszont egyik sem biztosít gyors és állandó minőségű hálózati elérést.

A probléma megoldására jött létre az IEEE 1905 munkacsoport [10], akik célul tűzték ki egy olyan szabványos megoldás létrehozását, amely kompatibilis a fentebb felsorolt technológiák mindegyikével és képes elfedni a felsőbb (layer 3) rétegek számára a köztük lévő különbségeket. A szab-

Hálózati			
IEEE 1905 Absztrakciós réteg			
MAC (IEEE 802.3)	MAC (IEEE 802.11)	MAC (IEEE 1901)	MAC (MoCA)
Fizikai (IEEE 802.3)	Fizikai (IEEE 802.11)	Fizikai (IEEE 1901)	Fizikai (MoCA)

Fig. 1: IEEE 1905 réteg architektúra

vány létrejött, aktuális verziója az IEEE 1905.1a-2014 [2] és több multipath megoldás jellemzőjét magában hordozza. Architekturálisan az 1905.1 absztrakciós réteg (AL) a különböző fizikai és adatkapcsolati rétegek fölött foglal helyet és az LLC (Logical Link Control) számára egységes MAC-ként (Medium Access Control) van jelen, elrejtve az alatta lévő heterogén hálózatot. Egyetlen EUI-48 MAC címet használ, az erre érkező és erről elküldött kereteket az AL-ben helyet foglaló továbbításért felelős entitás (forwarding entity) képezi le az alárendelt interfészekre. A protokoll képes felderíteni

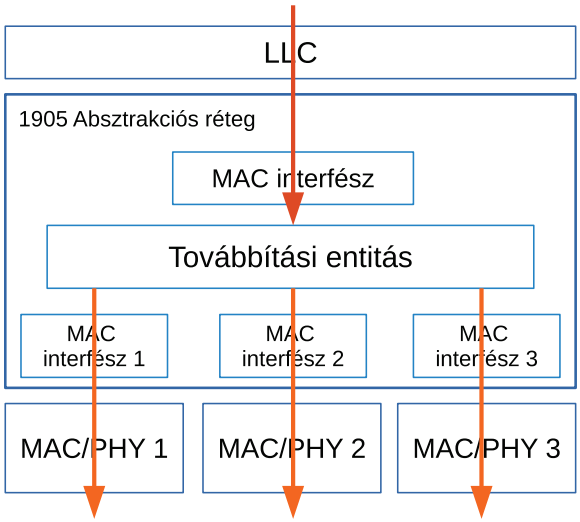


Fig. 2: IEEE 1905 kerettovábbítás vázlatos működése

a hálózati topológiát, a linkeken sebesség méréseket hajt végre, így találja meg a hálózat szűk keresztmetszeteit és esetleges hibás konfigurációkat. A felderített hálózaton egy egészet lefedő ütközési tartományt hoz létre, az így kombinált hálózat jobban lefedi a lakást. A felhasználó számára egységes konfigurációt tesz lehetővé (egy gombnyomásos beállítást minden IEEE 1905 kompatibilis eszköznek kötelezően implementálnia kell), elfedve a különböző technológiák eltérő, specifikus beállításait. A végpontok között megtalálható több útvonal közül választhat többféle költség alapján, előnyben részesítheti az alacsonyabb energiafogyasztású útvonalat, beállítástól függően viszont egyszerűen választhatja a gyorsabb útvonalat is. Az útvonalak konkurens használatával a forgalmat felosztathatja közöttük, elérve így a teljes hálózat maximális átbocsájtóképességét, aggregálva a különböző útvonalakhoz tartozó linkek sebességét. A forgalmat átirányíthatja másik útvonalra is, attól függően, hogy az aktuális útvonal hibaaránya mekkora, kielégíti-e a QoS (Quality of Service) beállításokat. Kritikus tartalmakat duplikálva, több útvonalon is továbbíthat ezzel növelve az átvitel megbízhatóságát, ekkor a túloldal, ha valamelyik útvonalon megkapja a tartalmat, az esetleges máshonnan érkező duplikátumokat egyszerűen eldobja. Képes gyors váltást eszközölni abban az esetben, ha az aktuálisan használt link megszakad, ekkor a forgalmat áttereli egy másik, még élő útvonalra, mindezt a felsőbb rétegek számára észrevétlen módon.

A protokoll hatékony működéséhez szükséges a bridge eszközök jelenléte a hálózatban. Az 1905.1 terminológiában ezek azok az eszközök, amelyek kettő vagy több médiumon is képesek adattovábbításra és alkalmasak a valamelyik hálózati interfészükön érkező kereteket egy másik interfészen tovább-

bítani. Egy televízió képes lehet bridge-ként működni MoCA és powerline közegek között, egy Wi-Fi router a vezetékes és vezeték nélküli Ethernet között továbbá a teljes lefedettséghez kell még egy eszköz ami a vezetékes Ethernet és a powerline között teremti meg az átjárást. Ebből a példából (feltéve hogy minden eszköz csak két médiumon képes kommunikációra) bármelyik bridge eszközt kihagyva nem jön létre egy teljes lefedést biztosító 1905.1 hálózat, hanem két diszjunkt és kisebb lefedettségű 1905.1 hálózat jön létre. Természetesen fontos kiemelni, hogy a lehető legjobb lefedettséghez minél több médiumon kommunikálni képes bridge eszközre van szükség. A szabvány még friss és bár ígéretes, a támogató eszközök elterjedése még a jövőben esedékes, amennyiben a piac vevő lesz a technológiára.

III. MPTCP

Az OSI protokoll hierarchia magasabb rétegében, a transzport rétegben működik a MultiPath Transmission Control Protocol (MPTCP). A protokollt az RFC 6824 [3] specifikálja részletesen, megtervezésekor [11] a kezdetektől fogva figyelembe vették a mai Internet jellemzőjét, miszerint két becsatlakoztatott eszköz között több lehetséges útvonal is jelen van. Ahhoz, hogy alkalmas legyen a jelenleg használatos TCP leváltására, szükséges hogy visszafelé kompatibilis legyen vele, és működjön minden olyan környezetben, ahol a TCP is. Abban az esetben, ha valamelyik fél nem támogatja az MPTCP-t, a kapcsolat visszadegradálódik hagyományos TCP kapcsolatra. További nehézség, ha a különböző útvonalak különböző middlebox-okkal (NAT/PAT, tűzfal, proxy, stb.) vannak ellátva, amik módosítják a TCP fejléc mezőit, ezzel ellehetetlenítve a többutas kapcsolat felépülését még akkor is ha a hálózati erőforrások megengednék ezt [12]. A protokoll

<table>
<tr><td colspan="3">Alkalmazás</td></tr>
<tr><td colspan="3">MPTCP</td></tr>
<tr><td>TCP
(alfolyam)</td><td>...</td><td>TCP
(alfolyam)</td></tr>
<tr><td>Hálózati</td><td>...</td><td>Hálózati</td></tr>
<tr><td>Adatkapcsolati</td><td>...</td><td>Adatkapcsolati</td></tr>
<tr><td>Fizikai</td><td>...</td><td>Fizikai</td></tr>
</table>

Fig. 3: MPTCP réteg architektúra

létrehozásakor figyelembe kellett venni az erősen heterogén hálózati környezeteket ahol a TCP már jól működik. Szerverparkok több gigabites, redundáns útvonalait kell hatékonyan kihasználnia, de mobiltelefonos környezetben, nagyságrendileg eltérő késleltetéssel rendelkező Wi-Fi és 3G-s útvonalak közti gyors váltást és linkaggregációt is képesnek kell lennie megvalósítani. A protokollnak mára van Linux kernel implementációja [13], az Apple iOS is implementálta bizonyos alkalmazásaihoz és léteznek módosított kernelek bizonyos Android operációs rendszert használó telefonokhoz (valamint egyes nagyobb gyártók már implementálták saját telefonjaikhoz pl. Samsung Galaxy S6-os nyílt implementáció [14]). Az MPTCP torlódás szabályozási eljárásai, utankénti stratégiái olyanok kell legyenek mint az egy utas hagyományos TCP stratégiái, annak érdekében, hogy ne boruljon fel az egyensúly olyan utakon, amiket egyidejűleg TCP is használ. Viszont másik oldalról, az algoritmus olyan kell legyen, hogy az utak között a lehető legnagyobb hatékonysággal ossza meg a forgalmat, abban az esetben ha valamelyik úton nagy torlódás jelenik meg se csoportosítsa át a forgalmat a kevésbé torlódott útvonalra, hogy esetleg ott okozzon torlódást. Az MPTCP aktuális kernel implementációja négy különböző torlódásvezérlési stratégiát tartalmaz [15], [16], [17], [18]. Ez is mutatja, hogy nincs általánosan érvényes, legjobb torlódásvezérlési algoritmus, helyzettől függ, hogy hol melyik a nyerő.

Mobilos környezetben, az utak eltérő technológiái eltérő átviteli karakterisztikákat mutatnak. A Wi-Fi útvonal például egy alapvetően gyors, de ingadozó minőségű átvitelt tesz lehetővé, a 3G mobilinternet egy viszonylag állandó, de magasabb késleltetésű utat ad, az LTE pedig a 3G-hez képest lényegesen kisebb késleltetésű és nagyobb sebességű kapcsolatot tesz lehetővé. Ebből adódóan a különböző utakon kiküldött csomagok felcserélődhetnek (tipikusan fel is cserélődnek, ezzel problémákat okozva a magasabb rétegek működési hatékonyságában [19]), így egy sorszámozási stratégia kidolgozására is szükség volt. Egyes middlebox-ok érzékenyek arra, ha a TCP sorszámok nem sorban jönnek (megpróbálnak újraküldést kérni, eldobják őket) [12], emiatt nem használhatók a TCP alfolyamok utankénti (kihagyásos) sorszámozással a csomag sorrend kizárólagos meghatározására. Ebből az okból az MPTCP bevezet egy saját, második szintű sorszámozást is, ami segít meghatározni hogy a hagyományos TCP sorszámok a teljes átküldendő adat sorban következő melyik bájtokat tartalmazzák, innentől kezdve az alfolyamok sorszámai lehetnek folytonosak.

Több mérés is megerősíti, hogy az MPTCP nagyon jól veszi a valós környezetek akadályait. Datacenter-es mérések igazolják [20], hogy a redundáns útvonalakat a protokoll nagyon jól kihasználja és hatékony linkaggregációt tud végrehajtani ezeken. Egy mérés [21] szerint 6 darab, egyenként 10 gigabites kapcsolat sebességét sikeresen összegzi 51.8 Gbit/s-má, az első öt aktív útvonalig közel lineárisan, a hatodiknál már kisebb hatásfokkal. Más mérések [22] mobilos környezetben igazolják, hogy a Wi-Fi és 3G közötti váltás is nagyon jól lekezelhetővé válik transzport rétegben. Több működési mód is támogatott (mobil eszközök esetében lényeges az energiagazdaságosság is). A protokoll használhatja valamelyik útvonalat backup útvonalként, ezt explicit módon speciális TCP flaggel jelzi a túloldal felé, ilyen esetben mindaddig az „olcsóbb" interfészen folyik az adattovábbítás, amíg azon él a kapcsolat, a másik útvonalon csak a háromutas kézfogás zajlik le. Ez a váltás a sebességben megmutatkozik, hiszen az ablakméretet fel kell növelni a váltás után a másik úton. A másik, költségesebb módszer, hogy az adatátvitel mindkét útvonalon folyik gyorsabb váltást tud eredményezni, hiszen az ablakméret már konvergált a másik útvonalon.

IV. MPT KOMMUNIKÁCIÓS KÖRNYEZET

Az MPT kommunikációs környezet architektúrája a „GRE in UDP" IETF draft [5] specifikációján alapszik. Az UDP

tunneling technológiát széles körben alkalmazzák különböző applikációs környezetekben. A GRE in UDP egységesíteni kívánja ezen tunneling technológiák protokoll stack-jét és PDU formátumát. Egy logikai tunnel interfészen a kommunikációs partnerek közvetlen szomszédként láthatják egymást, elfedve a tunnel interfész alatti hálózati infrastruktúrát. Az MPT ezt a gondolatot viszi tovább oly módon, hogy a tunnel interfész alatt lehetőséget nyújt több fizikai interfész alkalmazására és ezáltal többutas kommunikáció megvalósítására. Lehetővé teszi egy kommunikációs viszony esetén több út használatát, elkülönítve a kommunikációs viszony végpontjait az egyes hálózati interfészektől, a tényleges végpontnak magát a gépet tekintve. Az MPTCP-től több ponton is koncepcionális eltérés látható: az MPT működése a hálózati rétegben biztosított, így az alkalmazás tetszőleges transzport rétegbeli protokollt használhat. Az alkalmazások által a tunnel interfészen

Fig. 4: MPT réteg architektúra

Alkalmazás (tunnel)		
TCP/UDP (tunnel)		
IPv4 / IPv6 (tunnel)		
GRE in UDP		
UDP (fizikai)	...	UDP (fizikai)
IPv4/IPv6 (fizikai)	...	IPv4/IPv6 (fizikai)
Adatkapcsolati (fizikai)	...	Adatkapcsolati (fizikai)
Fizikai	...	Fizikai

keresztül küldött IP csomagokat az MPT szoftver „GRE in UDP" szegmensekbe csomagolja oly módon, hogy a fizikai továbbításhoz a rendelkezésre álló fizikai interfészeken megvalósított különböző útvonalakat használja. Ezáltal a tunnel interfészre érkező IP csomagok fizikai interfészekre történő dinamikus elosztása megvalósul, így biztosítva a többutas hálózati kommunikációt két végpont között. A tunnel interfész az alkalmazások számára ugyanúgy használható kommunikációra, mint egy fizikai interfész, legyen szó akár TCP, akár UDP protokollról. Megjegyzendő azonban, hogy a tunnel interfész alatt UDP protokoll működik, így ez a környezet önmagában nem biztosít újraküldési és forgalomszabályozási mechanizmusokat. Erre a célra az MPT szoftver egy kontroll interfészen keresztül biztosít csatlakozási és ezen keresztül forgalomszabályozási lehetőséget (pl. a tunnel forgalom elosztását a fizikai interfészek között).

Az MPT környezet az IPv4 és IPv6 protokollokat, illetve akár ezek kombinációját is támogatja a konfigurációban (például IPv6 használható a tunnel interfészen, és IPv4 a fizikai interfészeken, így az applikációk IPv6 protokollal kommunikálhatnak egymással egy IPv4-es infrastruktúra fölött). Az MPT környezet használatához szükség van az MPT szerver megfelelő konfigurációjára mindkét végponton (ld. [4]). Az MPT működési hatékonyságának vizsgálatára már számos kísérlet és publikáció született (ld. [23], [24], [25]), melyek azt bizonyítják, hogy az MPT hatékonyan képes az útkapacitások összegzésére. Ennek az aggregációs képességnek a maximumát vizsgálták a [26] cikkben.

Lehetőség van továbbá arra, hogy aktív kommunikációs viszony során dinamikusan változtassuk az utak számát, meglevőket kapcsoljunk le illetve fel, vagy akár lehetőség szerint újakat adjunk hozzá. Ezen tulajdonságot kihasználva könnyedén megvalósítható egy hálózati topológia esetén a több útra épülő redundancia a csomagveszteségek elkerülésére, kiváló példa ennek a funkciónak a hasznosságára vezeték nélküli utak esetén a layer 3 roaming (handover) megvalósítása. Ez a terület került vizsgálatra a [25] cikkben, terminálos (karakteres) felületű környezetben. Az elemzések azt mutatták, hogy a terminál kapcsolat folyamatosan fennáll Wi-Fi – 3G váltás esetén.

Ezt a munkát folytattuk, oly módon, hogy a Wi-Fi – 3G váltást egy (a kommunikációs késleltetésre érzékeny) videóstream átvitele közben vizsgáltuk. A vizsgálat során egy mindennaposnak mondható esetet reprodukáltunk: egy felhasználó egy notebookról csatlakozik egy videóstreamhez a helyi Wi-Fi hálózatot használva. A notebookon egy 3G hálózati interfész is található, aktív mobilinternet kapcsolattal. A felhasználó a videó fogadása közben valamilyen okból kénytelen lecsatlakozni a Wi-Fi hálózatról (pl. a bázisállomás hatótávján kívül kerül). A hagyományos egyutas kommunikációs környezetben ez a szituáció a videóstream leállásához vezetne, természetesen később a 3G kapcsolaton keresztül újraindulhat, de ehhez újra csatlakozni kell a videóstreamhez (intézze ezt akár az operációs rendszer, akár a felhasználó) és új kommunikációs viszonyt kell létrehoznia. Tehát ebben az esetben az adatvesztés és az ebből adódó problémák elkerülhetetlenek. Tanulmányunkkal azt vizsgáltuk, hogy az MPT szoftvercsomaggal megvalósított többutas kommunikációs környezetben ez a probléma hogyan orvosolható. A mérésekhez a következő tesztkörnyezetet állítottuk össze:

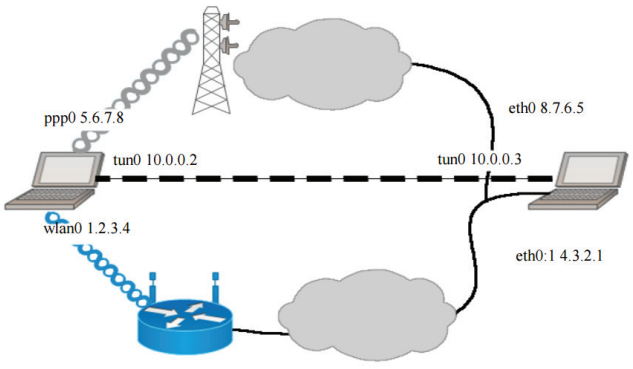


Fig. 5: A tesztekhez használt topológia

A videóstream funkcionalitást betöltő szerver (DELL Inspiron 3542 notebook: Intel Core i5-4210 (2700MHz) CPU, 8GB RAM (DDR3 1600MHz), 1000MB HDD (5400 RPM)) a Debreceni Egyetem Informatikai Karának épületében volt elhelyezve, a Gigabites hálózati interfészt használtuk mindkét út végpontjaként, két IP címet rendelve hozzá (az egyiket a fizikai interfészhez a másikat egy a fizikain létrehozott logikai interfészhez). A kliens számítógép (Intel Core i5-3210M (2500MHz) CPU, 8GB RAM (DDR3 1600MHz),

1000MB HDD (5400 RPM)) egy Wi-Fi és egy 3G interfészt használt, tehát két különböző internetszolgáltatón keresztül érte el a publikus világot. A tanulmány során két különböző működési környezetet vizsgáltunk: az egyik a Wi-Fi út tervezett lekapcsolása (pl. gyenge Wi-Fi jelszint esetén, még a kapcsolat megszakadása előtt), míg a másik a váratlan lekapcsolás, amely egy előre nem látható hálózati hibát hivatott szimulálni. Ez utóbbit a Wi-Fi bázisállomás internetkapcsolatának megszüntetésével értük el.

Mindkét működési környezetben TCP és UDP alapú RTP videóstreamelés működését is vizsgáltuk. Fontos kiemelni, hogy ezek a vizsgálatok a QoE-t (Quality of Experience) voltak hivatottak tesztelni. A méréseket több, független környezetben és többszöri alkalommal eltérő napszakokban is megismételtük. A 3G mobilinternetes út minden esetben a T-Mobile hálózatán keresztül valósult meg. A Wi-Fi út viszont a különböző helyszíneken eltérő paraméterekkel került megvalósításra: az első tesztkörnyezetben a Wi-Fi kapcsolat a Debreceni Egyetem Informatikai Karának épületén belül volt (egy hop távolságra a szervertől), így a késleltetések nagyon alacsonyak voltak és kis szórással rendelkeztek (8-12ms). A második tesztkörnyezetben az egyik hazai szolgáltató szélessávú elérését használtuk Debrecenen belül, Wi-Fi router-el megosztva. A harmadik tesztkörnyezet Budapesten került kiépítésre, egy munkakörnyezetben terhelt bérelt vonalas előfizetés Wi-Fi elérésén keresztül. A vizsgálataink minden helyszínen (azaz a Wi-Fi késleltetéstől függetlenül) homogén eredményeket mutattak: A tervezett leállás esetén egyik mérés esetén sem tapasztaltunk csomagvesztést. Ebben a környezetben a problémás szituáció a Wi-Fi interfész felkapcsolásakor jelentkezett, mivel ekkor (a Wi-Fi útvonal gyorsabb volta miatt) csomagsorrend átrendeződés volt tapasztalható. TCP alapú stream esetén a Wi-Fi út tervezett lekapcsolásával a streamben egyáltalán nem jelentkezett semmilyen probléma (sem kép vagy hanghiba, sem megszakadás). Az RTP stream esetén egy kicsivel rosszabb a helyzet, egy észrevehető képugrás (képtorzulás) volt megfigyelhető a le- és a felkapcsoláskor is (melynek oka a 3G interfész lényegesen nagyobb késleltetése és alacsonyabb sebessége). Ez a képhiba rövid idő (néhány másodperc) alatt helyreállt. Váratlan leállás során az MPT szoftver keepalive mechanizmusa [27] érzékelte a kommunikáció megszakadását, ez minden esetben valamennyi időt igényel, ennek eredményeképpen a videóban képmegállás illetve hangkiesés volt tapasztalható, melynek mértékét a keepalive üzenetek gyakorisága befolyásolta.

V. Jövőbeli tervek

Az MPT rendszer továbbfejlesztéséhez kapcsolódóan a vizsgálataink során nyert tapasztalatok két fejlesztési területet jelölnek ki:

A méréseink azt mutatták, hogy az utak eltérő sebességéből és késleltetéséből adódó csomagátrendeződés problémákat okozhat (pl. képtorzulás) a csomagvesztés mentes környezetben is. A GRE in UDP fejrész sorszám mezőjét alkalmazva a vevő oldalon egy pufferezés kialakításával lehetőség nyílik a csomagok GRE sorszám szerinti sorbarendezésére, ezáltal biztosítva, hogy a tunnel interfész sorrendhelyesen kapja meg a csomagokat abban az esetben is, ha a fizikai továbbítás során ez nem volt biztosított.

Az Android mobiltelefonos operációs rendszer 2015 második negyedévére a teljes piaci részesedés 82.8%-ával rendelkezett [28], ezzel magasan a legelterjedtebb a többi közül. A népszerűségét köszönheti a rengeteg rá elérhető alkalmazásnak. A Google jó fejlesztőeszközöket és jól dokumentált API-t (Application Programming Interface) ad a fejlesztők kezébe. A környezet rengeteg lehetőséget ad a rendszer hálózati programozása felé is. Az Android 5.1 operációs rendszer lehetőséget nyit a hálózati interfészek párhuzamos (együttes) használatára. Kísérleti tesztprogramjaink azt mutatják, hogy az itt rendelkezésre álló eszközök segítségével az MPT szoftver valamennyi funkcionalitása megvalósítható ezen a platformon, root jogosultság illetve kernelmódosítás nélkül is. Ezen kedvező feltételekre alapozva tervezzük a közeljövőben az MPT Android-os implementációját.

VI. Összefoglalás

A cikkben áttekintettük az aktuális többutas kommunikációs technológiákat. Részletesen ismertettük az MPT környezetben végzett videóstream átvitellel kapcsolatos kutatási eredményeinket. A tanulmányaink azt mutatják, hogy az MPT környezet jól alkalmazható Wi-Fi – 3G váltás (handover) esetén a problémamentes videóstream átvitel megvalósítására.

References

[1] “Cisco Visual Networking Index: Global Mobile Data Traffic Forecast Update, 2014–2019,” 2015.

[2] “IEEE Standard for a Convergent Digital Home Network for Heterogeneous Technologies Amendment 1: Support of New MAC/PHYs and Enhancements,” *IEEE Std 1905.1a-2014 (Amendment to IEEE Std 1905.1-2013)*, pp. 1–52, Feb 2015.

[3] A. Ford, C. Raiciu, M. Handley, and O. Bonaventure, “TCP Extensions for Multipath Operation with Multiple Addresses,” Internet Requests for Comments, RFC Editor, RFC 6824, January 2013, http://www.rfc-editor.org/rfc/rfc6824.txt. [Online]. Available: http://www.rfc-editor.org/rfc/rfc6824.txt

[4] B. Almási, “MPT - Multipath Communication Library.” [Online]. Available: http://irh.inf.unideb.hu/user/almasi/new/index.php/projektek/19-mpt-library

[5] E. Crabbe, L. Yong, X. Xu, and T. Herbert, “GRE-in-UDP Encapsulation,” Working Draft, IETF Secretariat, Internet-Draft draft-ietf-tsvwg-gre-in-udp-encap-07, July 2015, http://www.ietf.org/internet-drafts/draft-ietf-tsvwg-gre-in-udp-encap-07.txt. [Online]. Available: https://tools.ietf.org/html/draft-ietf-tsvwg-gre-in-udp-encap-07

[6] “HomePlug Alliance.” [Online]. Available: http://www.homeplug.org/

[7] “IEEE Standard for a Convergent Digital Home Network for Heterogeneous Technologies Amendment 1: Support of New MAC/PHYs and Enhancements,” *IEEE Std 1905.1a-2014 (Amendment to IEEE Std 1905.1-2013)*, pp. 1–52, Feb 2015.

[8] “Multimedia over coax alliance.” [Online]. Available: http://www.mocalliance.org/

[9] “Multimedia over Coax Alliance, MoCA 2.0.” [Online]. Available: http://www.mocalliance.org/MoCA2/index.htm

[10] “IEEE Convergent Digital Home Network Working Group home page.” [Online]. Available: http://grouper.ieee.org/groups/1905/1/

[11] C. Raiciu, C. Paasch, S. Barre, A. Ford, M. Honda, F. Duchene, O. Bonaventure, and M. Handley, "How Hard Can It Be? Designing and Implementing a Deployable Multipath TCP," in *Proceedings of the 9th USENIX Conference on Networked Systems Design and Implementation*, ser. NSDI'12. Berkeley, CA, USA: USENIX Association, 2012, pp. 29–29. [Online]. Available: http://inl.info.ucl.ac.be/publications/how-hard-can-it-be-designing-and-implementing-deployable-multipath-tcp

[12] M. Honda, Y. Nishida, C. Raiciu, A. Greenhalgh, M. Handley, and H. Tokuda, "Is It Still Possible to Extend TCP?" in *Proceedings of the 2011 ACM SIGCOMM Conference on Internet Measurement Conference*, ser. IMC '11. New York, NY, USA: ACM, 2011, pp. 181–194. [Online]. Available: http://doi.acm.org/10.1145/2068816.2068834

[13] "Linux Kernel implementation of MultiPath TCP." [Online]. Available: https://github.com/multipath-tcp/mptcp

[14] "Samsung Open Source Release Center, MPTCP enabled Galaxy S6 source code." [Online]. Available: http://opensource.samsung.com/reception/receptionSub.do?method=sub&sub=F&searchValue=SM-G925S

[15] M. Xu, Y. Cao, and E. Dong, "Delay-based Congestion Control for MPTCP," Working Draft, IETF Secretariat, Internet-Draft draft-xu-mptcp-congestion-control-02, July 2015, https://tools.ietf.org/html/draft-xu-mptcp-congestion-control-02. [Online]. Available: https://tools.ietf.org/html/draft-xu-mptcp-congestion-control-02

[16] A. Walid, Q. Peng, J. Hwang, and S. Low, "Balanced Linked Adaptation Congestion Control Algorithm for MPTCP," Working Draft, IETF Secretariat, Internet-Draft draft-walid-mptcp-congestion-control-03, July 2015, https://tools.ietf.org/html/internet-drafts/draft-walid-mptcp-congestion-control-03. [Online]. Available: https://tools.ietf.org/html/draft-walid-mptcp-congestion-control-03

[17] R. Khalili, N. Gast, M. Popovic, and J.-Y. L. Boudec, "Opportunistic Linked-Increases Congestion Control Algorithm for MPTCP," Working Draft, IETF Secretariat, Internet-Draft draft-khalili-mptcp-congestion-control-05, July 2014, http://www.ietf.org/internet-drafts/draft-khalili-mptcp-congestion-control-05.txt. [Online]. Available: https://tools.ietf.org/html/draft-khalili-mptcp-congestion-control-05

[18] D. Wischik, C. Raiciu, A. Greenhalgh, and M. Handley, "Design, Implementation and Evaluation of Congestion Control for Multipath TCP," in *Proceedings of the 8th USENIX Conference on Networked Systems Design and Implementation*, ser. NSDI'11. Berkeley, CA, USA: USENIX Association, 2011, pp. 99–112. [Online]. Available: http://dl.acm.org/citation.cfm?id=1972457.1972468

[19] L. Eggert and G. Fairhurst, "Unicast UDP Usage Guidelines for Application Designers," Internet Requests for Comments, RFC Editor, BCP 145, November 2008, http://www.rfc-editor.org/rfc/rfc5405.txt. [Online]. Available: http://www.rfc-editor.org/rfc/rfc5405.txt

[20] C. Raiciu, S. Barre, C. Pluntke, A. Greenhalgh, D. Wischik, and M. Handley, "Improving Datacenter Performance and Robustness with Multipath TCP," *SIGCOMM Comput. Commun. Rev.*, vol. 41, no. 4, pp. 266–277, Aug. 2011. [Online]. Available: http://doi.acm.org/10.1145/2043164.2018467

[21] C. Paasch, G. Detal, S. Barré, F. Duchéne, and O. Bonaventure, "The fastest TCP connection with MultiPath TCP," 2014. [Online]. Available: http://multipath-tcp.org/pmwiki.php?n=Main.50Gbps

[22] C. Paasch, G. Detal, F. Duchene, C. Raiciu, and O. Bonaventure, "Exploring Mobile/WiFi Handover with Multipath TCP," in *Proceedings of the 2012 ACM SIGCOMM Workshop on Cellular Networks: Operations, Challenges, and Future Design*, ser. CellNet '12. New York, NY, USA: ACM, 2012, pp. 31–36. [Online]. Available: http://doi.acm.org/10.1145/2342468.2342476

[23] B. Almási and S. Szilágyi, "Investigating the Throughput Performance of the MPT Multipath Communication Library in IPv4 and IPv6," *Journal of Applied Research and Technology*, Submitted.

[24] B. Almási and S. Szilágyi, "Multipath FTP and stream transmission analysis using the MPT software environment," *Int. J. of Advanced Research in Computer and Communication Engineering*, vol. 2, no. 11, pp. 4267–4272, 2013.

[25] B. Almási, "A Solution for Changing the Communication Interfaces between WiFi and 3G without Packet Loss," *Proceedings of 37th International Conference on Telecommunication and Signal Processing, Berlin, Germany*, pp. 73–77, 2014.

[26] G. Lencse and Á. Kovács, "Advanced Measurements of the Aggregation Capability of the MPT Network Layer Multipath Communication Library," *International Journal of Advances in Telecommunications, Electrotechnics, Signals and Systems*, vol. 4, no. 2, pp. 41–48, 2015.

[27] B. Almási, M. Kósa, F. Fejes, R. Katona, and L. Püsök, "MPT: a Solution for Eliminating the Effect of Network Breakdowns in case of HD Video Stream Transmission," 2015, Submitted.

[28] "Smartphone OS Market Share, 2015 Q2," 2014. [Online]. Available: http://www.idc.com/prodserv/smartphone-os-market-share.jsp

Paraméterbecslés 802.11ad rendszerekben

Csuka Barna és Kollár Zsolt
Szélessávú Hírközlés és Villamosságtan Tanszék
Budapesti Műszaki és Gazdaságtudományi Egyetem
1111 Budapest, Egry József utca 18.
E-mail: cs.barna86@gmail.com, kollar@mht.bme.hu

***Kivonat*—Cikkünkben a viszonylag újszerű, 2012-ben véglegesített IEEE 802.11ad szabványt mutatjuk be. Ez a szabvány teljesen új alapokra helyezi a vezeték nélküli átvitelt a jelenleg még ritkásan használt 60 GHz-es frekvenciasávban 2 GHz-es sávszélesség alkalmazása mellett. Célunk, hogy bemutassuk röviden a szabvány történetét és az átvitel során alkalmazott csomagok felépítését. Ismertetjük az átviteli láncban fellépő jelentősebb hibaforrásokat és azok hatását. Továbbá a kommunikációs rendszer alapsávi ekvivalens modelljét használva szimulációkon keresztül megoldásokat javaslunk a különféle hibák becslésére és kiküszöbölésére.**

***Kulcsszavak*—WiGig, IEEE 802.11ad, nagysebességű vezeték nélkül adatátvitel, paraméterbecslés**

I. Bevezetés

Az utóbbi évek fejlesztéseit megvizsgálva az látható, hogy egyre inkább az önállóan működő ún. okoseszközök felé halad a világ. Ezek a készülékek egymással szorosan együtt működnek vagy egymással laza függőségben lehetnek. Ez a viszony azonban az előző generációs eszközöknél tapasztaltakkal szemben semmiképpen sem szigorúan hierarchikus összekötést, láncolatot takar. Példának okáért korábban egy nyomtatóval csak egy számítógép segítségével, annak irányításával lehetett nyomtatni. Ezzel szemben ma már elterjedtek az önálló készülékek, amikre vezeték nélküli hálózaton kereszül nem csak PC-ről, hanem akár telefonról, tabletről is küldhetünk állományokat, illetve flashmemóriát is tud kezelni.

Ez a fejlődés magával hozta azt is, hogy egyre újabb és újabb adatátviteli eljárások jelentek meg vezeték nélküli átvitelre, így született meg az infravörös átvitel, a Bluetooth és az IEEE 802.11 szabvány különböző megoldásai. Időközben az átvinni kívánt adatmennyiség is növekedett (nagyfelbontású videók streamelése, másolása), így sorra jelentek meg az egyre gyorsabb átviteli eljárások: USB, FireWire, HDMI, USB 3.0 stb.; ám ezek elsősorban vezetékes technológiák.

Jelenleg az tapasztalható, hogy a mobil okoseszközök robbanásszerűen betörtek a mindennapokba, viszont az átviteli megoldásokkal egyenlőre csak próbáljuk utolérni a fejlődést. A fent felsorolt átviteli megoldások ugyan jók, viszont a csatlakozónak és magának a meghajtó áramkörnek is van helyigénye. Utóbbi azonban eléggé limitált pl. egy telefon esetében, ezért a legtöbb esetben csak egy jack- és egy adatátviteli – ami egyben töltő is – csatlakozó áll rendelkezésünkre. A hardveres nehézségeken túl a mobilitási igény miatt is cél a vezeték nélküli megoldások javítása. A fejlesztések két irányba folynak: egyik oldalon az alacsony átviteli sebességű, egyszerű megoldások állnak, amelyekkel sok kliens szolgálható ki egyszerre (pl. MiWi, ZigBee); míg másik oldalon a cél a lehető legnagyobb átviteli sebesség elérése és több kliens egyidejű kiszolgálása. Ilyen a vezeték nélküli HDMI átvitel (WirelessHD), vagy a cikkünk témájául szolgáló WiGig, ami az IEEE 802.11ad szabványát takarja.

Következő fejezetekben a 802.11ad szabványt mutatjuk be. Először a II. részben egy rövid történeti kitekintést adunk, hogy honnan indult a protokoll fejlesztése, és hogy hol tart ma. Ezt követően a III. fejezetben ismertetjük a rendszer főbb jellemzőit. A IV. részben bemutatjuk, hogy milyen átviteli hibaparaméterekkel kell foglalkozni az üzenetküldés során, és azok becslésére javaslunk eljárásokat. Ezután az V. részben bemutatjuk az elvégzett szimulációkat. Végül pedig összefoglaljuk az elért eredményeket és lehetséges továbbfejlesztési irányokat mutatjuk be röviden a VI. részben.

II. Az IEEE 802.11ad története

II-A. Előzmények

Az IEEE 802.11-es szabványcsoportja [1] a helyi, vezeték nélküli hálózatokat (WLAN) írják le, melyeket eredetileg időleges adatátvitelre terveztek egy fő hálózati eszköz és több végpont között viszonylag nagy lefedettség biztosítása mellett. Ezzel pont ellentétesek a mai követelmények: folyamatos kapcsolatra van szükségünk, amely Gbps nagyságrendű sebességet garantál nagyon kicsi késleltetéssel, továbbá a hálózatnak elosztott jellegűnek kell lennie, hogy két kliens közvetlenül egymással is tudjon kommunikálni (pl. televízió irányítása mobileszközzel, vagy videó streamelése egyik készülékről a másikra).

Az IEEE sorozatos fejlesztésekkel (802.11b, 802.11g, 802.11n) illetve a használt frekvenciasávok bővítésével (2,4 GHz, 5 GHz) próbálta követni az igényeket. Azonban azt ezek a javítások sem tudták kiküszöbölni, hogy telítődtek a frekvenciasávok, egyre több eszköz osztozik ugyanakkora sávszélességen. Emiatt az elérhető elméleti adatátviteli sebességnek csupán a töredéke áll rendelkezésre egy-egy felhasználó számára, így nem garantálható a folyamatos, nagysebességű adatátvitel.

Az analóg és a digitális elektronika fejlődése mára lehetővé tette az eddig nem használt frekvenciasávok kihasználását kommunikációs célokra úgy, hogy a modulok integrálhatóak mobileszközökbe is. Éppen ezért a WiGig rendszer (és az azt leíró IEEE 802.11ad szabvány [2]) a jelenleg még üres 60 GHz-es frekvenciasávot használja ki a nagy átviteli sebesség elérése végett. Ezzel a megoldással lehetővé válik, hogy a WiGig egymagában helyettesítsen minden vezetékes és vezeték

nélküli adatátvitelt az egy légtérben, 10-15 méteres körön belül található eszközök között. Ezáltal egy újabb lépéssel közelebb kerülhetünk az eszközeink konvergenciájához, vagy ahogy ma közkeletűen nevezik: a dolgok internetéhez [3].

II-B. Első lépések és a szabványosítás

A WiGig kidolgozását, az első prototípusok tervezését és gyártását a Wilocity végezte el a Qualcomm Atheros-szal közösen. Előbbi céget 2007-ben, Kaliforniában alapította a korábban az Intelnek is dolgozó mérnökök egy csoportja, majd a Qualcomm Atheros felvásárolta 2014-ben. A projektbe egyre több nagy fejlesztő- és gyártócég kapcsolódott be, így a WiGig nem csak a szabványt jelenti, hanem az azt fejlesztő csoportosulás, a Wireless Gigabit Alliance rövidítése is. A teljesség igénye nélkül a fenti kettőn kívül a következő cégek tagjai az együttműködésnek: Agilent, Apple, Dell, Huawei, Intel, Microsoft, Nokia, Panasonic, Rohde & Schwarz, Sony, Samsung, Texas Instruments.

A szervezet a szabvány kidolgozása során szorosan együttműködött az IEEE 802.11 szabványt gondozó Wi-Fi Alliance szövetséggel, és az új szabványt a meglévő, az IEEE 802.11 által megadott keretrendszerekhez igyekeztek igazítani. Az első verzió, a WiGig 1.0 2009-ben jelent meg, amit 2011-ben követett a végleges, 1.1-es változat. A közös munka eredményeként az IEEE adoptálta a WiGig 1.1-es szabványt felülírva a fejlesztés alatt álló 802.11ad kiegészítést. Így akadálytalanul sikerült a WiGig-et az IEEE 802.11 szabványcsalád részévé tenni úgy, hogy csak a fizikai és a MAC-réteget tekintve jelent változást a specifikációban, a többi réteg változatlan maradt az IEEE korábbi szabványainak megfelelően. Az IEEE 802.11ad véglegesen az IEEE 802.11-2012-es szabványverzióba került bele [2], ezt követően a WiGig szervezet bele is olvadt a Wi-Fi Allience szervezetbe, így azóta együtt felügyelik, tesztelik, minősítik az azóta megjelent termékek szabványnak való megfelelését [3].

II-C. Jelenlegi helyzet

Azután, hogy a WiGig az IEEE 802.11 része lett, definiálták az FST (Fast Session Transfer) protokollt, amely segítségével az arra felkészített, háromsávos készülékek hardverszinten váltanak a Wi-Fi korábbi, 2,4/5 GHz-es és a WiGig 60 GHz-es frekvenciái között. Ezzel a megoldással az eszközök folyamatosan kapcsolódnak az elérhető legjobb hálózathoz, és az átkapcsolás idejére se szakad meg a kapcsolat.

A szabvány megoldásai és tulajdonságai miatt (nagy sebesség, kicsi késleltetés) lehetővé vált, hogy különböző adaptációs rétegek segítségével a korábban csak vezetékes összeköttetések vezeték nélküli változatát is definiálják. Jelenleg a következő megoldások érhetőek el: Wireless Bus Extension (PCIe alapján), Wireless Serial Extension (USB alapján), Wireless Display Extension (HDMI és DisplayPort alapján) és Wireless SDIO Extension (SDIO alapján). Ezekkel a protokollokkal teszi lehetővé a szabvány a vezetékek kiváltását úgy, hogy a készülékek felé marad a korábbi csatolófelület (pl. USB, HDMI), csupán annyi a változtatás, hogy a csatlakozó másik végén a vezeték helyett egy 802.11ad adóvevő található, ami kapcsolódni tud bármilyen másik 802.11ad klienshez, és ezáltal egy másik eszközhöz.

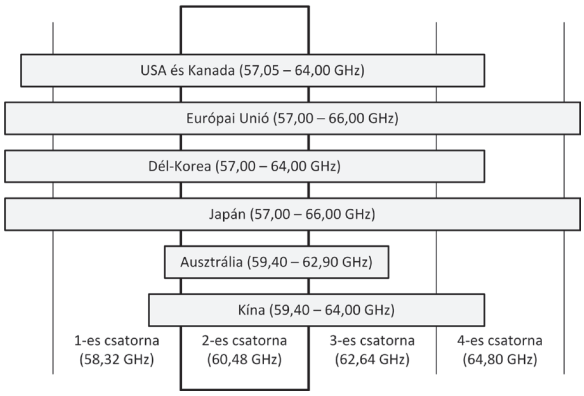
Az első chipcsalád a Wilocity 6100-as családja volt, amit a 6200-as és a 6300-as követett, utóbbi már mobileszközök számára készült. A 6100-as és a 6200-as család megoldásaira épített a Dell, amikor megjelent az első, kereskedelmi forgalomban kapható, 802.11ad-vel szerelt laptop, a Dell 6430u és a hozzá tartozó D5000-as vezeték nélküli dokkoló. A dokkolóval egyetlen egy WiGig linken keresztül a laptophoz csatlakoztatható három USB 3.0, egy HDMI-, egy DisplayPort- és egy LAN-készülék, továbbá sztereó audió összeköttetés is a rendelkezésre áll.

2014-ben a Qualcomm Atheros jóvoltából megjelent az első, mobileszközökbe szánt alaplap, a Snapdragon 810, amely már WiGig-hálózathoz is tud csatlakozni. A már megjelent megoldások mellett folyamatosan fejleszti számos gyártó (pl. Intel, Panasonic, Samsung, Sony) a saját eszközét, melyek megjelenése az elkövetkező időszakban várható [3].

III. AZ IEEE 802.11AD TULAJDONSÁGAI

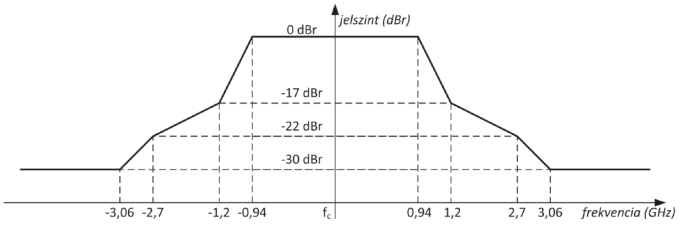
III-A. Fizikai réteg

A WiGig alapsávi sávszélessége 2 GHz, amit 60 GHz-es vivőfrekvenciára kevernek fel, vagyis milliméteres hullámtartományban üzemel a rendszer, melynek az elméleti maximális átviteli sebessége 7 Gbps. A Nemzetközi Távközlési Egyesület Rádiókommunikációs Osztálya (ITU–R) az 57-66 GHz közötti frekvencián 4, egyenként 2,16 GHz sávszélességű csatornát rögzített. Ezen csatornák közül a 2-es csatorna érhető el az egész világon, aminek 60,48 GHz a sávközepi frekvenciája, ezért ez lett alapértelmezettként rögzítve a WiGig szabványban, ahogy az 1. ábra is mutatja. Az egy csatornára vonatkozó, betartandó spektrummaszkot a 2. ábra mutatja [4], [5].



1. ábra. A 60 GHz-es tartományban rögzített csatornák és sávközépi frekvenciájuk

A WiGig a fizikai rétegében az átvitel során egy- illetve többvivős OFDM átvitelt alkalmaz. Az egyvivős átvitel során az alapsávi jel egy komplex, digitálisan modulált jel, ahol szabvány a következő három modulációt támogatja: BPSK, QPSK és 16-QAM. A többvivős, OFDM átvitel esetén az alapsávi jelet, az ún. OFDM-szimbólumot 512 pontos inverz Fourier-transzformációval állítják elő, és az átvitel során 355 alvivőt alkalmaznak. Az alvivőkön a következő modulációk használhatóak: QPSK, SQPSK, 16-QAM és 64-QAM [2], [3].



2. ábra. Egy csatornához tartozó spektrummaszk a sávközéphez viszonyítva

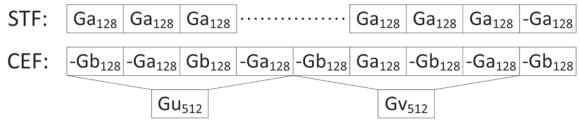
III-B. Keretformátum

Az adatcsomag, ahogy a 3. ábra mutatja, a következő részekből épül fel egyvivős átvitel esetén: preambulum, fejléc, adatrész illetve a nyalábformálási adatok. Cikkünkben paraméterbecslési célokból csupán a preambulumot használjuk fel, így a következőkben csak ennek a bemutatására szorítkozunk [2].

Preambulum		Fejléc	Adatrész	Nyalábformálási rész
STF	CEF			

3. ábra. A 802.11ad csomagszerkezete

A preambulum az ún. rövid szinkronizációs részt (Short Training Field – STF) és az ún. csatornabecslő részt (Channel Estimation Field – CEF) tartalmazza. Ezek a mezők építőelemei komplemens Golay-szekvenciák [6], [7], melyeknek négy típusa van: Golay-A (Ga), Golay-B (Gb), illetve ezek negált változatai. Ebben az esetben a szekvenciák 128 pont hosszúságúak, melyeket ezért a következőképpen jelölünk: Ga_{128}, $-Ga_{128}$, Gb_{128} és $-Gb_{128}$. Ezek a szekvenciák könnyen generálhatóak az ún. Golay-generátorral (A. függelék), ahol az $N = 128$ esetre meg is adtuk a szükséges beállítási paramétereket.



4. ábra. Az STF és a CEF felépítése

Az STF és a CEF mezők felépítése a 4. ábrán látható. Az STF egy periódikus jel, amelyben 20-szor ismétlődik meg a Ga_{128}, majd egy $-Ga_{128}$ mezővel zárul. Ezzel szemben a CEF nem periódikus, viszont komplemens szekvenciákat tartalmaz, így ez a rész a csatornabecsléshez használható fel. Három fő része van: egy Golay-U (Gu_{512}), egy Golay-V (Gv_{512}) és egy $-Gb_{128}$ szekvencia.

Az OFDM átvitel esetén alkalmazandó fejléc csak kismértékben tér el az előbb ismertetett megoldástól. A Gu_{512} és a Gv_{512} fordított sorrendben szerepel, a Golay-V megelőzi a Golay-U szekvenciát. Ezért a cikkben bemutatott paraméterbecslési eljárások változtatások nélkül alkalmazhatóak OFDM átvitel esetén is.

IV. Átviteli paraméterek becslése

Egy általános átviteli lánc felépítése az 5. ábrán látható. Az adó oldalon a moduláció után az alapsávi jeleket egy D/A-konverterrel analóg jellé alakítjuk, majd pedig a helyi oszcillátor segítségével felkeverjük a vívősávi frekvenciára. A valós és a képzetes részt ezt követően összegezzük és egyetlen rádiós jelként adjuk ki az erősítést követően.

A rádiós csatorna jellemezhető egy idővarián, frekenciaszelektív átviteli függvénnyel. A kiadott jel ezen a rádiós csatorán áthaladva lineáris torzítást szenved. Ezen torzított jelhez a különféle környezeti hatások miatt zaj is adódik. A vevő oldalon a torzított, zajjal terhelt jelet a helyi oszcillátor segítségével az alapsávra lekeverjük és egy aluláteresztő szűrővel szűrjük. Ezt követően tudjuk digitalizálni, demodulálni és feldolgozni a kapott jelmintákat. Az adatok helyes értelmezése végett kompenzálni kell az átviteli út során fellépő hibákat; ehhez az azokat jellemző paramétereket meg kell becsülni. Ebben a cikkben, a következő részben a teljesség igénye nélkül a következő paraméterbecslésekkel foglalkozunk: időzítés, frekvencia és fázis offset illetve az átviteli csatorna karakterisztikája. Ezen paraméterek becsléséhez az STF és a CEF mezőket fogjuk használni.

IV-A. Időzítés

A legelső probléma, amivel szembesülünk a vevő oldalon, hogy meg kell állapítanunk, hogy mikor kezdődik az adás, mikortól használhatóak a vett jelminták demoduláláshoz. A vevőantenna mindig vesz valamennyi környezeti háttérzajt, ezenfelül az utána következő erősítőknek, szűrőknek is van valamekkora elektromos zaja. Ezen okok miatt a jelfeldolgozó egység bemenetén mindig lesz valamilyen mérhető és nagyságú jel, legyen az akár hasznos, akár nem. Ezt kiküszöbölendő, szükségünk van egy időzítési megoldásra, algoritmusra, amely megadja, hogy hányadik vett minta után kezdődött az adás, ami után már a demodulációt elkezdhetjük.

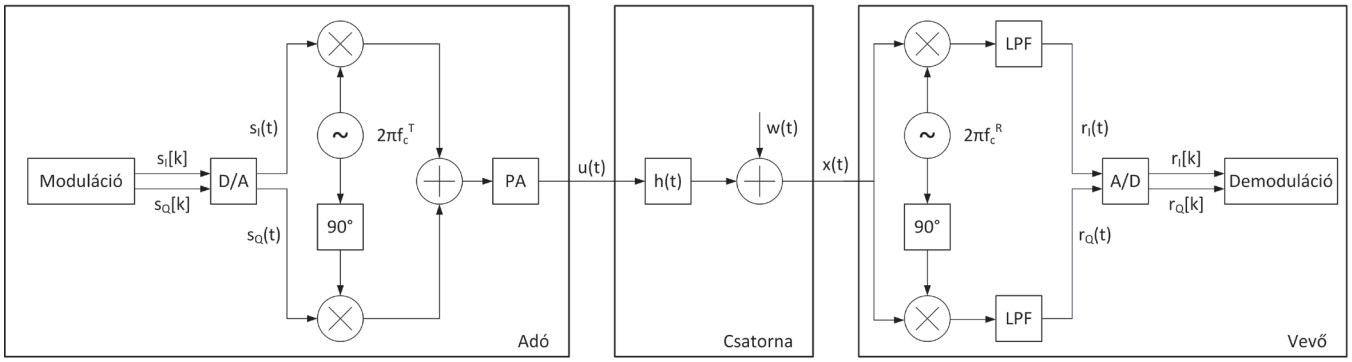
Erre a problémára a következő megoldást az STF felhasználásával adta Li et. al [8]. Kihasználva az STF periódikus felépítését, a következő, (1) és (2) korrelációs kifejezések kiszámolhatóak mindegyik k-adik bejövő jelmintára:

$$P[k] = \sum_{l=1}^{6}\sum_{m=0}^{M-1} x[d+m]\cdot\overline{x}[d+m+M\cdot l] + \\ + \sum_{m=0}^{M-1} x[d+m+M]\cdot\overline{x}[i+m+6M], \quad (1)$$

$$R[k] = \sum_{l=1}^{6}\sum_{m=0}^{M-1} |x(d+m+M\cdot l)-x(d+m)|^2 + \\ + \sum_{m=0}^{M-1} |x(d+m+6M)-x(i+m+M)|^2, \quad (2)$$

ahol $x[k]$ az k-adik mintát jelenti, M az egységnek választott blokkhosszúság és $\overline{x}$ pedig x komplex konjugáltja. Ha ezt a két korrelációt kiszámoltuk, akkor ezek segítségével már képezhető a következő időzítési metrika (3), amely segítségével a csomag kezdete meghatározható:

$$S[k] = \frac{|P[k]|}{R[k]} \quad (3)$$



5. ábra. Egy általános átviteli lánc felépítése

A (3) kifejezést a számítások legvégén még a könnyebb kezelhetőség végett célszerű normálni, és így egy olyan korrelációs függvény adódik, amely ott veszi fel a maximális értékét, ahol a küldött csomag elkezdődik. Minden másik helyen közel nulla az értéke, vagyis a függvény egy nagyon meredek felfutású csúcsot tartalmaz, amely csúcsának helye megadja a megfelelő időzítést.

IV-B. Frekvencia offset becslése

Mivel az adó és a vevő egymástól független, ezért az oszcillátoruk frekvenciái eltérhetnek egymástól. Amennyiben tényleg nem egyenlőek, akkor a vett konstellációs pontok elkezdenek forogni a komplex síkon $\omega_c = 2\pi\Delta f_c$ szögsebességgel, ahol a Δf_c az oszcillátorok frekvenciáinak a különbsége.

Arra az esetre, amikor az egymás után küldött adatblokkok tartalma megegyezik, Moose bebizonyította [9], hogy a Δf_c maximum likelihood becslője megadható a következő kifejezéssel, mivel a csatorna átviteli függvénye állandónak tekinthető:

$$\Delta\widetilde{f}_c = (1/2\pi)\arctan\left\{\frac{\left(\sum_{k=0}^{N-1}\Im\left[X_{2k}\overline{X}_{1k}\right]\right)}{\left(\sum_{k=0}^{N-1}\Re\left[X_{2k}\overline{X}_{1k}\right]\right)}\right\}, \qquad (4)$$

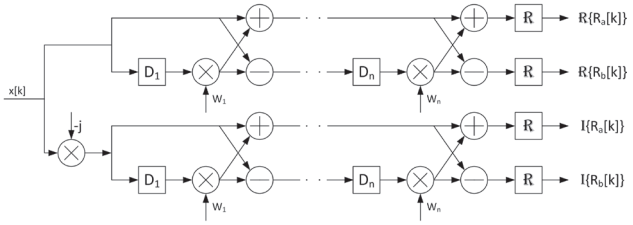
ahol X_{1k} és X_{2k} a k-adik binjei két, egymás után fogadott adatblokk spektrumának, és X komplex konjugáltját pedig $\overline{X}$ jelöli. A vett jelminták spektrális felbontás elvégezhető a klasszikus gyors Fourier-transzformációval (FFT).

IV-C. Csatornakarakterisztika becslése

A rádiós csatorna frekvencia-szelektivitása miatt kompenzálni kell a vett jelet a demoduláció előtt. Ennek elvégzéséhez becsülni kell az átviteli csatorna karakterisztikáját azzal a megkötéssel, hogy egy 802.11ad csomag átvitele alatt a csatornát konstansnak tekintjük. Ebben a részben a preambulum alapján becslő eljárásokat mutatjuk be: először a korrelációs eljárást, majd pedig az FFT-alapú becslőt, legvégén pedig egy újszerű, ún. bővitett, FFT-alapú eljárást.

IV-C1. Korrelációalapú becslés. Ahogy a III-B. részben említettük, a CEF részei egymás komplementerei illetve negáltjai, ami a korrelációszámítás szempontjából kedvező tulajdonság. Először a CEF-nek a Ga_{128} illetve a Gb_{128} szekvenciákkal vett korrelációját kell kiszámolni. A kapott értékeknek éles csúcsuk van; amennyiben ezeket a korrelált sorozatokat megfelelő pozícióba toljuk, akkor a csúcsok egymásra lapolódnak. Ebben az esetben – mivel a sorozatok értékei egymás negáltjai – a csúcs körüli értékek zérusok lesznek, kivéve magát a csúcsot. Így a csúcs, és az utána következő pontok tartalmazzák a csatorna impulzusválaszát.

Ezen korrelációk kiszámítására hatékony megoldást nyújt a Golay-korrelátor. Ennek több változata ismert, jelen esetben az ún. gyors Golay-korrelátort használjuk [10]. Az eljárás ugyanúgy működik, mint a generátor (lásd (5) egyenlet), csupán a kezdeti értékek (a_0 és b_0) nem a Kronecker-féle deltafüggvényt veszik fel, hanem az aktuálisan fogadott adatot.



6. ábra. C–FGC sematikus felépítése

Ahhoz, hogy komplex csatorna esetén is helyes eredményt kapjunk, módosítani kell a gyors Golay-korrelátort, hogy külön-külön tudja számolni a valós és a képzetes résszel vett korrelációkat. A bővített korrelátor, a komplex gyors Golay-korrelátor (C–FGC) felépítése az 6. ábrán látható.

IV-C2. Fourier-transzformáció alapú becslés. Másik eljárás a csatorna becslésére a Fourier-transzformáció. Ebben az esetben a Gu_{512} és a Gv_{512} becsült spektrális megfelelői $\widetilde{X}_U$ és $\widetilde{X}_V$ lesznek, melyek FFT-vel számíthatóak. Az elméleti, ideális spektrális értékek legyenek X_U és X_V, melyeket korábban már kiszámítottunk és elmentettük referencia értékekként.

A csatorna átviteli függvénye ($\widetilde{H}$) a következő módon fejezhető ki: $\widetilde{H}_U = \widetilde{X}_U/X_U$ és $\widetilde{H}_V = \widetilde{X}_V/X_V$. Ez a két becslés összevethető, és a becsült átviteli függvény $\widetilde{H}_{ch}$ megadható $\widetilde{H}_U$ és $\widetilde{H}_V$ átlagolásával, végül a $\widetilde{H}_{ch}$-ből inverz diszkrét Fourier-transzformációval kiszámítható a csatorna impulzusválasza. Fontos megjegyzés: az X_U spektrumának az első binje zérus, ezért a nullával való osztás elkerülése végett a $\widetilde{H}_U$ első binjét interpolálni kell, vagy pedig a $\widetilde{H}_V$ megfelelő értékével kell helyettesíteni.

A CEF a Gu_{512} és a Gv_{512} elemeken kívül egy lezáró $-Gb_{128}$ elemet is tartalmaz (III-B. rész), ami ciklikusságot visz a CEF-be, ugyanis mindkét, 512 hosszú blokk egy $-Gb_{128}$-cal kezdődik. Így lehetővé válik, hogy a Gv_{512} spektrumát újra kiszámoljuk az utolsó $-Gb_{128}$ beérkezése után is (lásd 7. ábra). A $-Ga_{128}$ egy hasonló, ciklikusságot biztosító blokk, így amennyiben az STF utolsó, $-Ga_{128}$ mezőjét is felhasználjuk, akkor a Gu_{512} spektruma is kétszer számolható ki. Ezzel a megoldással kettő helyett négy becsült spektrumot lehet átlagolni, a hozzáadott, 128 hosszú blokkok egynegyed részt függetlenek a korábbi adatoktól, így a becslés varianciáját csökkenthetjük ezzel a megoldással.

7. ábra. Bővített FFT

IV-D. Fázishiba

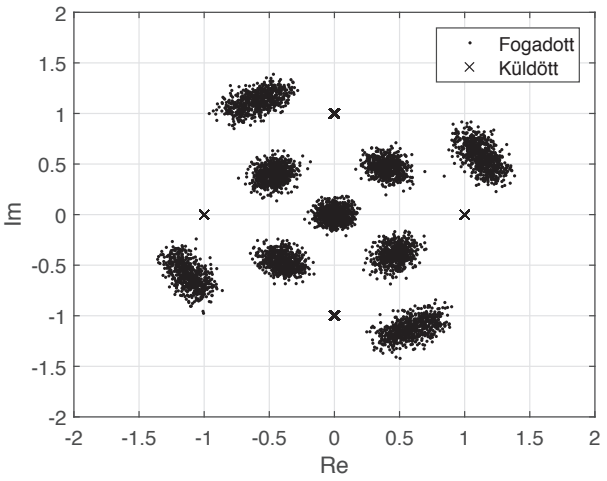
A két független oszcillátor nem csak a frekvenciában különbözik egymástól, hanem különböző fázishelyzetük is van. Ezt a konstans fáziskülönbséget a csatornakorrekció korrigálni tudja. Azonban a frekvenciahiba nem konstans, valamennyi maradó hiba az átvitel folyamán végig marad, ami azt jelenti, hogy a csomag fogadása közben is lesz egy kicsi fázisváltozás. Ennek kiköszöbölésére és detektálására az adatcsomag is tartalmaz Golay-szekvenciákat periódikusan, azonban ezzel a hibával jelen cikk keretein belül nem foglalkozunk.

V. Szimulációs környezet

Ebben a részben azokat a szimulációkat mutatjuk be, amelyeket egy, a Matlab-ban készített, a 802.11ad-t modellező keretrendszer segítségével készítettünk. A szimulációk során küldött csomagok minden esetben egy teljes adatcsomagot tartalmaztak (3. ábra), és csak az alapsávi átviteli modellt alkalmaztuk, a fel- és lekeverést nem szimuláltuk (vö. 5. ábra). A preambulumnál, a fejlécnél és az adatrésznél is $\pi/2$-BPSK modulációt alkalmaztunk a szabványnak megfelelően [2]. Ez a modulációs séma kétlépcsős: előbb a szükséges fázisbillentyűzéssel kell modulálni az adatokat, majd ezután egy folyamatos, $\pi/2$-es forgatást kell alkalmazni a modulált adatokon. Ennek hatását a 8. ábra is mutatja: BPSK moduláció esetén csak a $(-1,\ 1)$ pontok alkotnák a konstellációt, azonban a forgatás miatt az elrendezés kiegészül a $(-1\mathrm{j},\ +1\mathrm{j})$ pontokkal is.

V-A. Az átviteli csatorna szimulációja

A csatorna különböző torzító hatásait a 8. ábra mutatja be; látható, hogy a $\pi/2$-BPSK moduláció négy pontja hány különböző pontra képződött le az átviteli lánc különböző hatásai miatt. A IV-A. fejezetben rámutattunk, hogy nagyon fontos a vétel kezdetének pontos időzítése. Ezt szimulálandó, az átvitel során az adatok elé véletlenszám generátor által adott számú nullát szúrtunk be, ennek köszönhető, hogy a 8. ábrán az origó környezetében is jelentek meg pontok.

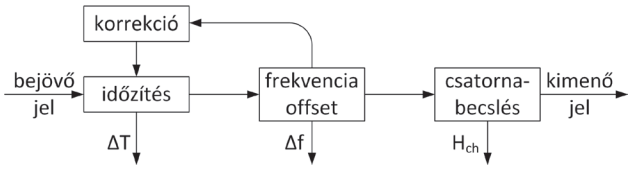


8. ábra. A küldött és a fogadott adatok elhelyezkedése a komplex síkon

A IV-B. részben bemutattuk, hogy az adó és a vevő oszcillátorok függetlensége mit okoz. Ezt egy rögzített szögsebességű forgatással szimuláltuk, ebben az esetben pl. 10 ppm-es beállítással. Ennek hatása látható a 8. ábrán, a pontok origó középpontú körív mentén „haladnak előrefelé". A csatornatorzítás hatásának bemutatására (IV-C. fejezet) egy véletlenszerű átviteli karakterisztikával rendelkező szűrővel szűrtük az adatokat. Jelen esetben a csatorna impulzusválasza kéttagú volt, emiatt látható a 8. ábrán, hogy a négy konstellációs pont két pontnégyesre képződött le. Végül pedig a csatorna zajának hatásának szimulálására véletlen, normál eloszlású, zérus középértékű komplex számokat adunk a küldött adatokhoz, emiatt mosódnak el ovális alakban a fogadott pontok a 8. ábrán, ahol a zaj szórását az éppen beállított, 20 dB-es jel-zaj viszonyból (SNR) kaptuk meg.

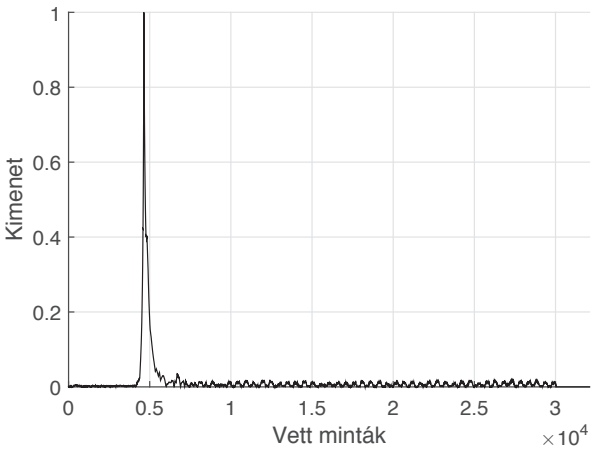
V-B. Az átviteli paraméterek meghatározása

A paramétereket becslő rendszer felépítése a 9. ábrán látható: először meghatározzuk az adás kezdetét, majd az oszcillátorok közötti frekvenciakülönbséget. Ezt követően a kapott becslők alapján végrehajtjuk a korrekciót, majd ezután újra elvégezzük ezeket a becsléseket. Amennyiben az itt kapott hibák nagysága adott hibahatáron belül van, akkor tovább lehet lépni a csatornabecslésre.

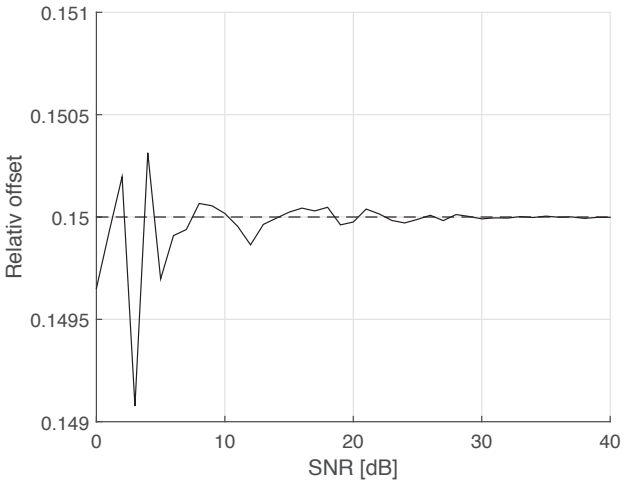


9. ábra. Paraméterbecslő-rendszer felépítése

Az időzítő blokk az (1)-(3) egyenletek által megadott kifejezéseket számolja ki, és a kapott $S[k]$ függvény a 10. ábrán látható alakot veszi fel. A jól detektálható, éles csúcs segítségével könnyen megállapítható, hogy mikortól vesz a vevő hasznos adatokat.



10. ábra. Az időzítő blokk kimenetén kapott $S[k]$ függvény



11. ábra. A frekvencia offset becslője az SNR függvényében

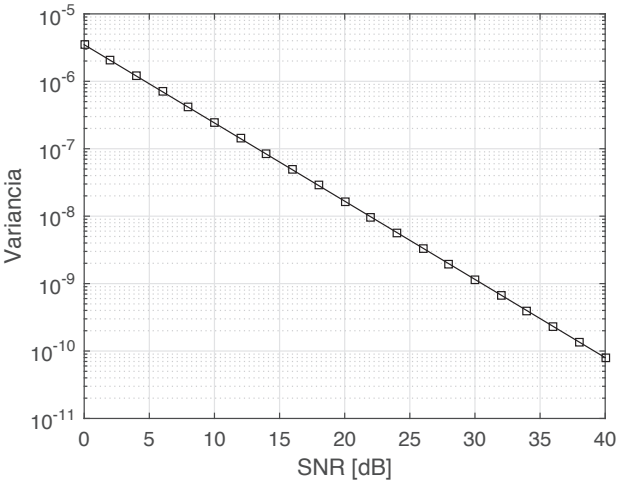
A frekvencia offset becsléséhez a (4) képlet használható fel, amely ugyan OFDM átvitel számára lett megadva, de módosításokkal a 802.11ad esetében is alkalmazható. Az X_1 és X_2 blokkoknak az STF két, egymás után fogadott Ga_{128} szekvenciájának kell lennie. További Ga_{128} blokkok felhasználásával – amíg van még fel nem használt ga – további becslések adhatóak az offset nagyságára. Az így kapott becslések átlaga megadja a tényleges becslőjét az offsetnek, az STF esetében 13 pár alkotható, vagyis 13 becslést lehet számolni. Fontos megjegyezni, hogy az első egy-két Ga_{128} blokkot nem célszerű használni, mivel azokat a kezdeti tranziensek torzíthatják.

A szimulációk során az SNR-t 0 és 40 dB között léptettük 2 dB-s közzel. Mindegyik SNR érték esetén 25 átvitelt vizsgáltunk, és mindegyikhez kiszámoltuk a $\Delta\widetilde{f}_c$-t. A kapott eredményeket átlagoltuk, illetve a varianciájukat is meghatároztuk, majd a kapott értékeket a 11. és a 12. ábrákon ábrázoltuk. A Δf_c 0,15-re lett beállítva a szimulátorban, az átviteli csatornát pedig ideálisnak tekintettük ebben az esetben.

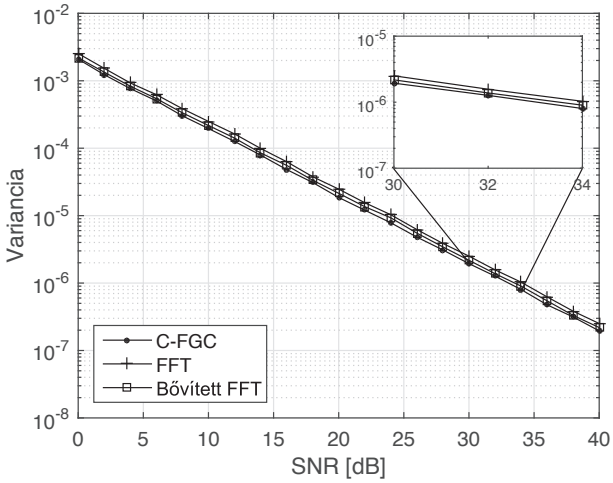
Az átviteli csatorna becslésére szolgáló két eljárást a IV-C. fejezetben mutattunk be. A korrelációs megoldást a C–FGC adja (lásd IV-C1. rész). A Fourier-transzformáció alapú megoldást FFT-vel számoltuk ki $N = 512$ pontos felbontás alkalmazásával (lásd IV-C2. rész).

Egy új megoldást is megvizsgáltuk, ami egy olyan, bővített számítási eljárás, melyben az STF utolsó $-Ga_{128}$ és a CEF utolsó$-Gb_{128}$ blokkjait is felhasználtuk. A számítás bővebb leírása a IV-C2. részben szerepel, illetve magának a bővítésnek az elvét a 7. ábra mutatja.

A szimulációk során a jel-zaj viszonyt, az SNR-t 0 dB-től 40 dB-ig növeltük 2 dB-es lépésközzel. Minden egyes SNR értékhez 25 átvitelt generáltuk, majd minden átvitelre kiszámoltuk a becsült impulzusválaszokat. A kapott eredményeket átlagoltuk, és a varianciát is kifejeztük. A szimulációk eredményét a 13. ábra mutatja. Az alkalmazott impulzusválasz – melynek a becslőjét kerestük – kéttagú volt, frekvenciahibát ebben az esetben zérusnak vettük.



12. ábra. A frekvencia offset becslőjének varianciája az SNR függvényében



13. ábra. A csatornabecslés varianciája az SNR függvényében

V-C. A kapott eredmények kiértékelése

A szimulációk elvégzése után a következő megállapítások tehetőek. Az időzítő eljárás nagy pontossága miatt (10. ábra) lehetővé válik az adatok pontos vétele, ezáltal pontos feldolgozása. A 11. és a 12. ábrákon látható, hogy a frekvencia offsetet jól tudjuk becsülni, az SNR növekedésével egyre pontosabb a becslés, és ezzel együtt egyre kisebb is lesz a varianciája. Ennek köszönhetően az oszcillátorok hibáit a pontos feldolgozáshoz szükséges hibahatáron belüli mértékre tudjuk csökkenteni.

A csatorna átviteli karakterisztikájának becslésére három eljárást is bemutattunk, a 13. ábra mutatja a kapott becslések varianciáit. A becslési metódusok egyformán hatékonyak, egyformán csökken a varianciájuk az SNR függvényében, csupán minimális különbség mutatkozik az eredmények között (13. ábra kinagyított része).

VI. Összefoglalás

A cikkünkben bemutattunk egy viszonylag új rendszert, a WiGig-et, amely jó megoldást kínál a ma használatos, vezeték nélküli rendszereknél tapasztalt problémákra (II-A. rész). Ugyanakkor azt is láthattuk, hogy annak ellenére, hogy a 802.11ad egy, már szabványosított protokoll [2], jelenleg még csupán csak korlátozottan érhetőek el ezen eljárást használó eszközök (II-C. fejezet). Ennek oka leginkább az, hogy 60 GHz-en üzemelő eszközöket, alkatrészeket nehéz előállítani, nincsenek bevált tervezési, gyártási eljárások, módszerek, ezért a fejlesztő cégek számos nehézségbe ütköztek, ütköznek.

A IV. fejezetben bemutattunk egy általános átviteli láncot, és megmutattuk, hogy a kiadott jeleket milyen hatások érik az átvitel során. Ezek a hatások különböző hibákat okoznak a vételi oldalon, de a hibák paramétereit meg tudjuk becsülni. A IV-A – IV-D. fejezetekben bemutattunk olyan becslési eljárásokat, amelyek a 802.11ad esetében is hatékonyan használhatóak. Végül pedig az V. fejezetben szimulációk segítségével is igazoltuk, hogy a bemutatott eljárások ténylegesen hatékonyak tudnak lenni.

A megkezdett munka folytatására két út mutatkozik. Egyrészt a becslő eljárásokat érdemes még tovább vizsgálni, azokat tovább fejleszteni, finomítani. Másrészt pedig egy teljesen elkészült szimulációs keretrendszer segítségével olyan adatcsomagok generálhatóak, amelyeket megfelelő jelgenerátorba töltve, fel- és lekeverők alkalmazásával az átvitel a fizikai valóságban is tesztelhetővé válik; a vételi oldalon a bemutatott eljárások verifikálhatóak.

Függelék A
Komplemens Golay-szekvenciák

Golay olyan bináris szekvenciákat mutatott be [6], amelyek páronként egymás komplementerei. Ez azt jelenti, hogy az autokorrelációjuk összege zérus, kivéve, ha maga az időparaméter, a k zérus.

Budišin adta meg a következő algoritmust [7], [10], mellyel a megfelelő, N hosszú Golay-szekvenciák – $a[k]$ és $b[k]$ – generálhatóak:

$$\begin{aligned} a_0[k] &= b_0[k] = \delta[k] \\ a_n[k] &= a_{n-1}[k] + W_n \cdot b_{n-1}[k - D_n] \\ b_n[k] &= b_{n-1}[k] - W_n \cdot b_{n-1}[k - D_n], \end{aligned} \tag{5}$$

ahol $\delta[k]$ a Kronecker-féle deltafüggvény, n a rekurziók száma. A W_n a szorzó együtthatókat tartalmazza, míg D_n a késleltető elemek hosszát adja meg.

Az $N = 128$ esethez a szabványnak megfelelő szekvenciák generálásához a következő értékek szükségesek a inicializáló vektorokhoz, $n = 7$ rekurziós lépést (mivel $\log_2 N = 7$) alkalmazva [2]:

$$\begin{aligned} \mathbf{W} &= [-1, -1, -1, -1, +1, -1, -1], \\ \mathbf{D} &= [1,\ 8,\ 2,\ 4,\ 16,\ 32,\ 64]. \end{aligned}$$

Hivatkozások

[1] „IEEE Standard for Information Technology – Telecommunications and information exchange between systems Local and Metropolitan Area Networks – Specific requirements Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications," *IEEE Std 802.11-2012 (Revision of IEEE Std 802.11-2007)*, Mar. 2012.

[2] „IEEE Standard for Information Technology – Telecommunications and information exchange between systems Local and Metropolitan Area Networks – Specific requirements Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications – Amendment 3: Enhancements for Very High Throughput in the 60 GHz Band," *IEEE Std 802.11ad-2012 (Amendment to IEEE Std 802.11-2012, as amended by IEEE Std 802.11ae-2012 and IEEE Std 802.11aa-2012)*, Mar. 2012.

[3] F. Plesznik, „Új generációs szélessávú vezeték nélküli adatátvitel," Diplomamunka, Budapesti Műszaki és Gazdaságtudományi Egyetem, 2015.

[4] *Wireless LAN at 60 GHz – IEEE 802.11ad Explained*, Agilent Technologies, 2013, URL: http://cp.literature.agilent.com/litweb/pdf/5990-9697EN.pdf (ellenőrizve: 2015. augusztus).

[5] *802.11ad – WLAN at 60 GHz – A technology introduction*, Rohde & Schwarz, 2013, URL: https://cdn.rohde-schwarz.com/pws/dl_downloads/dl_application/application_notes/1ma220/1MA220_1e_WLAN_11ad_WP.pdf (ellenőrizve: 2015. augusztus).

[6] M. J. E. Golay, „Complementary series," *IRE Trans. on Inf. Theory*, vol. 7, no. 2, pp. 82–87, Apr. 1961.

[7] S. Z. Budišin, „New complementary pairs of sequences," *Electronics Letters*, vol. 26, no. 13, pp. 881–883, June 1990.

[8] S. Li, G. Yue, X. Cheng, and Z. Luo, „A Novel and Robust Timing Synchronization Method for SC-FDE 60 GHz WPAN Systems," in *IEEE 14th International Conference on Communication Technology*, 2012, pp. 262–267.

[9] P. H. Moose, „A technique for orthogonal frequency division multiplexing frequency offset correction," *IEEE Trans. Commun.*, vol. 42, no. 10, pp. 2908–2914, Oct. 1994.

[10] S. Z. Budišin, „Efficient pulse compressor for Golay complementary sequences," *Electronics Letters*, vol. 27, no. 3, pp. 219–220, Jan. 1991.

Modern technológiákon alapuló otthoni felügyelő rendszer

Kalmár György[1], Balázs Péter[2]

[1] Szegedi Tudományegyetem, Képfeldolgozás és Számítógépes Grafika Tanszék
kalmargy@inf.u-szeged.hu

[2] Szegedi Tudományegyetem, Képfeldolgozás és Számítógépes Grafika Tanszék
pbalazs@inf.u-szeged.hu

Absztrakt. *A betörések, rablások megelőzése vagy megakadályozása a hétköznapi ember számára csak riasztó berendezés használatával lehetséges, amely viszont drága és beszereléséhez szakemberre van szükség. Kutatásunk célja egy olyan alacsony költségű, mindenki számára elérhető, könnyen beüzemelhető és használható, mobil alapú védelmi berendezés megalkotása, amely képes megtanulni, majd később felismerni a védelmezett terület tulajdonosának arcát, felhasználva a ma már olcsón beszerezhető új, illetve használaton kívül esett régi okostelefonokat. A kifejlesztett felügyelő rendszer jelenleg egy arcot képes megtanulni, majd azt a későbbi megjelenés esetén felismerni. Ismeretlen arc esetén MMS vagy e-mail üzenetet küld a felhasználójának az arcról készült képpel. A szakirodalomban már jól kiforrott módszereket megfelelően egy működő rendszerré integráló Android alapú alkalmazás lehetővé teszi arcról nagy felbontású képek nagyon gyors kimentését, ezzel teret adva arcfelismerő algoritmusok használatának. A cikkben a kezdeti elért eredményeinket ismertetjük.*

I. BEVEZETÉS

Napjaink egyik kiemelkedő társadalmi problémája a betörések, lopások, rablások megakadályozása. Habár széles körben elérhetők otthonilag használható riasztóberendezések, ezek beszerelése sokszor túl költséges egy család számára. Továbbá egy rövid kódon múlik biztonságuk, amelyet akkor is igényelnek hangos riasztás közepette, ha a valódi tulajdonos érkezik otthonába. Megállapítható tehát, hogy ezek a rendszerek sokszor alacsony tudásúak és kényelmetlenek. Ezzel szemben egy okostelefon, amely ma már mindenki számára a hétköznapok része, a kommunikáció és multimédia egész tárházát nyújtja egy tenyérnyi kis eszközben, magas processzor teljesítménnyel, sok memóriával, kamerával, mikrofonnal, internet hozzáféréssel, telekommunikációs funkciókkal.

Kutatásunk célja, egy olyan mindenki számára elérhető, olcsó, intelligens otthoni felügyelő rendszer kivitelezhetőségének a vizsgálata és megalkotása, amely képes megtanulni a rendszer használójának arcát, majd későbbi ismételt megjelenés esetén felismerni azt, ismeretlen személy esetén pedig riasztást küldeni MMS vagy e-mail üzenet formájában, amely tartalmaz egy képet az azonosítatlan arcról. Ezen rendszer elsősorban az elavulttá vált, sérült okostelefonok segítségével rendkívül kis költségigénnyel járulna hozzá a hétköznapi emberek biztonságához.

Ma már az okos eszközök nagyon nagy hányadán Android alapú operációs rendszer fut. Ez lehetővé teszi, hogy az egyes eszközök különbségeit a felsőbb rétegekben elrejtsük. Tehát egy elkészült, védelmi célokat biztosító alkalmazás az összes Android alapú eszközön futni képes. Ezen megfontolásból a megvalósításához mi is Android alapú eszközt választottunk és az elkészült alkalmazást is Java nyelven, Androidos környezetben fejlesztettük ki.

II. A MEGVALÓSÍTOTT RENDSZERRŐL ÁLTALÁNOSSÁGBAN

A jelen megvalósításban tárgyalt rendszer csak egy arc megtanulására képes. Ezen korlátozás kihatással van az alkalmazott módszerre. A mesterséges intelligencia megalkotásánál figyelembe kell venni, hogy a döntéshozás bináris jellegű, tehát megtanult arc vagy nem az szerepel egy képen.

Az arcfelismerés statikus képek esetén bár jól, de nem tökéletesen megoldott probléma. Esetünkben a felismerés nehézségéhez hozzájárul az is, hogy nem élhetünk azzal a feltételezéssel, hogy a felvétel az arcról szemből készül. Ezért már egy személy felismerésénél is adódnak nehézségek. Elsősorban az intraperszonális variancia, vagyis hogy egy személy saját magához képest mennyire változatos, jelenti a fő nehézséget. Akadályozó tényező továbbá, hogy esetünkben a felismerendő személyek mozognak, így csak homályos képeket készíthetünk, ha figyelembe vesszük, hogy az okostelefonok kamerája nem a legkifinomultabb optikai rendszer és nem is a gyors képkészítésre optimalizált.

A képfeldolgozási algoritmusokat az alkalmazás az OpenCV4Android nevű csomagon keresztül éri el, amely az OpenCV [1] Androidos környezetre előkészített változata. Az eljárások nagy része C++-ból fordított natív kód formájában kerül futtatásra a Java-s környezetből egy JNI (Java Native Interface) csatoláson keresztül, ezzel lényegesen felgyorsítva a végrehajtást.

A rendszerrel szemben elvárt funkciók:

- legyen képes egy olcsó okostelefonon is futni (minimális elvárás, hogy az rendelkezzen egy hátoldali kamerával, egy mikrofonnal, és képes legyen Android 4.0-s vagy újabb verziót futtatni)
- legyen képes egy arc megtanulására, ezt egy egy napos, felügyelt tanítási fázisban kell, hogy megtegye
- a tanítási fázis után képes legyen egy megjelenő arcról eldönteni, hogy az a megtanult arc (személy) vagy sem
- mindezen felismerő funkciókat a lehető leghatékonyabban és legmegbízhatóbban végezze

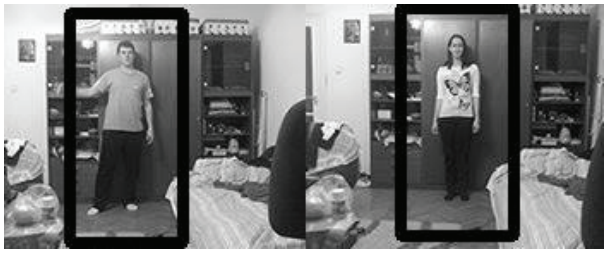
- minimalizálja a fals pozitív döntések számát, hiszen ez azt jelentené, hogy ismeretlen arcot ismertnek feltételezne
- ha az arc felismerése fizikai korlátok miatt nem is hajtható végre, akkor is próbáljon meg döntést hozni az illető kilétéről, ezt jelen megvalósításban egy hangalapú jelszóellenőrző algoritmus végzi el

A rendszer beüzemelésének kezdetekor az adott okostelefonra fel kell telepíteni az alkalmazást. Elindítva a programot, az egyből a tanulási fázisba lép. Ezen fázis során tanulja meg a később felismerni kívánt arcot, illetve a hangalapú jelszó azonosításhoz a referencia jelszót. A tanulás végeztével a rendszer egyből riasztó, felügyelő üzembe lép. Ez az üzemmód az alkalmazás leállításáig tart.

III. A RENDSZER ALAPKÖVEI

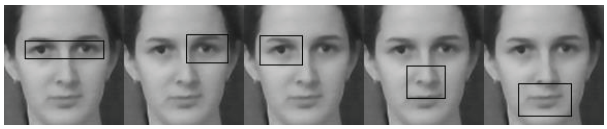
Az emberi alak megbízható és gyors felismerése az egyik, és talán a legfontosabb alappillére a felügyelő rendszernek. Ennek köszönhető, hogy a képen megjelenő személy detektálásra kerül, így a rendszer értesül arról, ha a megfigyelt térségben valaki tartózkodik. Jelenleg erre a célra, az OpenCV által is támogatott HOG leíróval működő módszert használjuk [2,3]. Ennek előnye, hogy nagyon megbízható, a fals pozitív értesítések száma minimális. Ugyanakkor igen műveletigényes, gyengébb processzoron és nagyobb felbontású képen futtatva több másodpercig is eltart az elemzés. Ennek kiküszöbölésére hoztuk létre a később tárgyalásra kerülő saját kamera kezelő osztályt, amely segítségével már a hardvertől érkező képek felbontását megfelelően alacsonyra tudjuk állítani ahhoz, hogy a HOG algoritmus szinte valós időben működhessen. Az emberi alak detektálásának működése során készült képek az 1. ábrán láthatóak.

Az emberi alak megtalálásán túl a rendszer másik elengedhetetlen eleme az arcok megbízható és gyors detektálása, tekintve, hogy a végső cél ezek megtanulása és felismerése. Még az emberi alak többnyire egy általános formának mondható, addig az arc nagyon változatos kinézetű is lehet. Az OpenCV az eddig publikált leggyorsabb és legrobusztusabb arcfelismerő eljárást támogatja. Ennek lényege, hogy úgynevezett Haar jellemzőket nyernek ki a képekből, majd ezekre építve gyenge osztályozókat hoznak létre, amelyeket egymás után fűznek (Haar cascade classifier) [4]. Az OpenCV ezen eljárásra támaszkodva, előre tanított osztályozókat biztosít, amelyek nem csak arc, de sok más objektum keresésére is alkalmasak [5]. Ennek köszönhetően választottuk az arcfelismerés módszeréül azt, hogy az arcról ún. mikrojellemzőket vonunk ki és ezeket hasonlítjuk össze a később megjelenő arc jellemzőivel. A mikrojellemzők konkrétan a száj, az orr, a bal szem, a jobb szem, és a szempár, pontosabban az ezekről készült képrészletek. Az arc és arcrészek detektálására példákat a 2. és 3. ábrán láthatunk.



1. ábra: Emberi alak felismerés

2. ábra: Arc detektálása



3. ábra: Arcrészek detektálása

IV. KIEGÉSZÍTŐ FUNKCIÓK

A felügyelő rendszer alapvető funkcióinak tárgyalása után a következőkben bemutatjuk a nem szükségszerű, de nagyon hasznos kisegítő eljárások működését. Ezek közé tartozik a pehelysúlyú mozgásérzékelő, a hangalapú jelszófeldolgozó, a dinamikus arckövető algoritmus és a saját kamera kezelő osztály.

Habár az emberi alak keresés nagyon hatékony a kis méretű képen, és így a feldolgozási sebességet is magasan lehet tartani, mégis indokolatlan egy ilyen bonyolultságú algoritmust futtatni minden képre. Ha ugyanis nincs mozgás a képen, akkor biztosak lehetünk benne, hogy ember nem sétált be a megfigyelt területre. Ezt kihasználva egy mozgásérzékelő algoritmus előszűrője lehet az emberi alak keresésnek, miszerint azt csak akkor érdemes futtatni, ha történt a képen jelentős változás. Nyilvánvaló, hogy ez az algoritmus sokkal egyszerűbb, kisebb processzor és memóriaigénnyel. A szakirodalom *background subtraction*, azaz háttér kivonás néven hivatkozik az olyan algoritmusokra, amelyek célja, hogy egy képen elkülönítsék a háttérhez és az objektumhoz (előtérhez) tartozó pixeleket. A legkiemelkedőbb módszerek ezen a téren valószínűségi háttér modelleket alakítanak ki. Az általunk használt, OpenCV-ben megvalósított módszer lényege, hogy minden pixel intenzitását egy tanító képhalmaz segítségével egy normális eloszlással adja meg. Vagyis minden pixelre meghatároz egy várható értéket és szórást, és az eloszlást normálisnak feltételezi (változó paraméterű Gauss görbékkel ír le minden pixelt). Innen ered a neve, Gaussian Mixture Model, vagyis Gauss görbék keverékéből álló modell [6]. Az OpenCV-ben szereplő verzió nagyon gyors és műveletigénye is alacsony ([7]). Működése közben készült

képeket a 4. ábrán figyelhetünk meg.

Az eddig tárgyalt funkciók célja, hogy lehetővé tegyék a felügyelő rendszer számára, hogy érzékelje az emberi jelenlétet és megpróbálhassa a személy arcát felismerni. Számos okból előfordulhat azonban, hogy az arc detektálása fizikai okokból eleve lehetetlen (pl.: rossz fényviszonyok, eltakart vagy a kamerának háttal forduló arc, gyors mozgás, amely miatt lehetetlen éles képet készíteni). Ilyen esetekben is valahogy meg kellene arról bizonyosodni, hogy a detektált emberi alak ismert archoz tartozik-e.

A probléma megoldásához a hang alapú interakciót választottuk. Az azonosítás egy előre megtanult jelszó ismételt kimondásával valósul meg. A tanulási fázis elején a felhasználó egy tetszőleges jelszót mondhat be a rendszerbe, amit az felvesz és megjegyez. Ha a későbbi éles működés során az arc felismerése sikertelen lenne valamilyen okból kifolyólag, akkor a rendszer egy hangjelzés segítségével „megkéri" a felügyelt területen tartózkodó személyt, hogy ismételje el a megtanult jelszót. Ezt felvételezi, majd kinyeri a felvételi idő beszéd részét. A referencia és az aktuálisan meghatározott beszéd régiók korrelációja alapján döntést hoz, hogy ugyanaz a szó hangzott-e el.

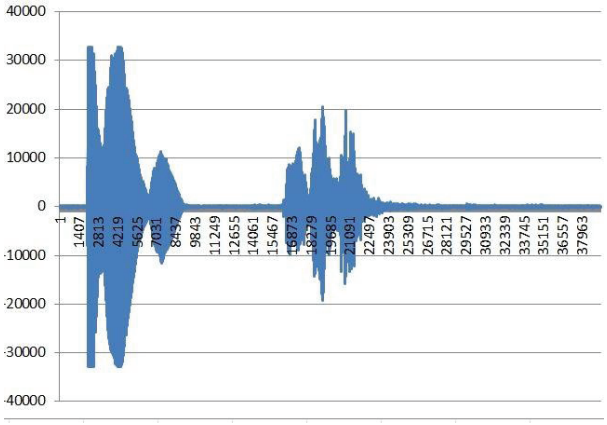
Az eszköz által felvett audio jel nagyon zajos lehet. Erre láthatunk példát az 5. ábrán. A jelen egyenirányítást majd szűrést hajtunk végre a következő szűrő segítségével (csúszóablakos, súlyozott, átlagoló szűrő):

$$f(x) = \frac{1}{2n+1} \sum_{i=-n}^{n} (n - |i| + 1) * y(x+i), x = n, \dots, L - n$$

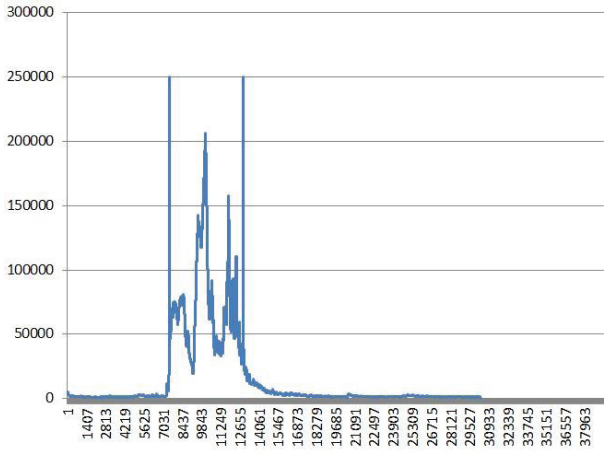
ahol *f(x)* a szűrt jel az *x* pozícióban, *n=N/2*, ahol *N* a csúszóablak szélessége. *L* az eredeti jel hossza, *y(x)* pedig az értéke az *x* pozícióban.

Az 5. ábrán látható jelalak elején megfigyelhető kiemelkedések a rendszer által kiadott hang miatt jelennek meg, ami mindig ugyanolyan hosszú, tehát eldobható ez a rész. Az 5. ábra jelének szűrt, kezdeti részt nem tartalmazó alakja a 6. ábrán figyelhető meg, amelyen a két kiemelkedő csúcs az algoritmus által meghatározott beszéd régió kezdő és végpontját jelöli. A beszéd régió jelen megvalósításban a leghosszabb, nem zajszinten lévő régió.

4. ábra: Mozgásérzékelés



5. ábra: Nyers hangfelvétel hullámformája



6. ábra: Szűrt jel, bejelölt beszéd régióval

Ha már rendelkezésre állnak a beszéd régiók, akkor ezeket kinyerve a referencia és az aktuálisan felvett hangból, kezdődhet az összehasonlítás. Ennek során az összehasonlítandó beszédrészt ugyanolyan hosszúra alakítjuk, mint a referencia jelszó hossza (nyújtás, majd interpoláció). Ez azért fontos, mert ugyanazt a szót többféle ritmusban is ki lehet mondani. Ha már egyforma hosszúak, normalizált keresztkorrelációval kiszámítjuk az egyezést a két hangalapú jelszó közt. Ha egy előre megadott küszöböt meghalad a hasonlóság mértéke, akkor a bediktált jelszót elfogadjuk, egyébként pedig elvetjük.

Már az előzőekben említés esett arról, hogy fontos és elengedhetetlen az arcról nagyon gyorsan, a lehető legnagyobb felbontással képeket készíteni. Ehhez szükségünk van egy olyan eljárásra, amely biztosítja, hogy ne a teljes képet kelljen átkutatni arc után. Ehhez két feltevéssel élhetünk:

1. Emberi alak fellelése után elegendő csak az emberi alak helyének felső részén keresni arc után egy adott méretű régióban.

2. Ha már találtunk arcot, valószínű, hogy a következő képen is ehhez a helyhez nagyon közel lesz az arc.

Ezeket kihasználva egy olyan módszert fejlesztettünk ki, amely nyomon tudja követni az arc helyzetét. Az eljárás egy növekvő méretű keresési ablakot használ. Az ablak mérete azzal arányosan nő, hogy hány képkocka óta nem volt arc az

adott ablakban. Ha ismét talál egy arcot, akkor az ablak középpontját áthelyezi az arc középpontjába és innen folytatja újra a növekedést, ha ismét eltűnne az arc. A 7. ábrán egy olyan sorozatot láthatunk, amelynek első képén az ablak az emberi alak detektálás után állítódott be. Majd a további képeken a mozgás hatására változó helyzetű és méretű ablak figyelhető meg.

Az alkalmazás szempontjából kritikussá vált, hogy elérhetőek legyenek a következő, kamerát érintő funkciók:

1. Változtatni lehessen a kamerából érkező kép felbontását.
2. Folyamatosan bekapcsolva lehessen tartani a felvétel során a kamerához tartozó LED vakut (éjjeli üzemmód).

Ezen funkciókat csak saját kamera kezelő osztály megírásával lehet elérni. Ennek megvalósítását követően a felbontás váltása hardver szinten gyorsan, kevesebb, mint egy másodperc alatt végbemegy, illetve folyamatos üzemben tudjuk működtetni a kamerához tartozó LED vakut.

7. ábra: Dinamikus arckövetés

V. A RENDSZER MŰKÖDÉSI FÁZISAINAK VIZSGÁLATA

A rendszer feladata, hogy megjelenő arcokat „ismert” vagy „ismeretlen” címkékkel lásson el. Ehhez azonban először egy felügyelt tanulási fázison kell átesnie, amely során elsajátíthatja a később felismerendő arcot. Ezt az alkalmazás első indításakor teszi meg. Maga a tanulás a később felismerni kívánt jellemzők kivonását, elmentését jelenti. Jelen megvalósításban ezek a leírók az arcról kivont mikrojellemzők: száj, orr, bal szem, jobb szem, szempár. Ez az öt kép kerül elmentésre egy referencia arcról, minden arcrész egy előre megadott egységes méretre skálázva. A rendszer ezen mikroképeket egy napig gyűjti, fontos, hogy ez az idő alatt csak a később felismerni kívánt felhasználó jelenjen meg a felügyelt területen.

A képkészítések során homályos, elmosódott képek sokszor készülnek. Ennek kiszűrése érdekében egy referencia helyett több kerül elmentésre minden arcrészről. A korrelációt zavarja a fény intenzitásának és irányának változása, ezért ezt kiküszöbölendő, a rendszer egy nap során, több alkalommal is készít referenciákat, időben elszeparálva. Ha egy lokális referencia gyűjtési időben már megvan a kellő számú minta minden mikrojellemzőről, akkor a rendszer tovább folytatja a képek gyűjtését, és az itt kivont jellemzőket azonnal össze is hasonlítja az éppen rögzített referencia jellemzőkkel. Egy személy adott arcrészét sajátjával hasonlítjuk össze. Ezen autokorrelációs értékekből sok készül el. Ezekből átlagot számolva egy dinamikusan meghatározott küszöbszintet állítunk elő a későbbi felismerés számára, melyet időben hozzárendelünk a referenciákhoz. Egy referenciacsoportot és a hozzá rendelt küszöbszinteket láthatjuk a 8. ábrán.

8. ábra: Rögzített referenciacsoport dinamikusan meghatározott küszöbszintekkel.

Amikor a tanulás véget ér, a felügyelő fázis kezdi meg munkáját. Ebben a módban, a rendszer folyamatosan, bizonyos időnként referencia betöltést hajt végre. Ez a dinamikus referencia betöltés. Ennek során az elmentett referenciákat végignézve dönt arról, hogy a betöltés idején (valószínűsíthetőleg) melyek azok, amelyek a legjobban illenek az akkori fényviszonyokhoz (intenzitás, irány).

A felügyelés során folyamatos mozgásérzékelés történik. Ha mozgás jelenik meg a képen, akkor a rendszer egyből emberi alak detektálásra vált át. Ha nem talál embert, akkor bizonyos idő után visszaáll mozgásérzékelésre. Ha talál, akkor a dinamikus arckövető eljárással arc után kezd kutatni. Ha nincs arc, akkor jelszót kér bemondásra, hiszen arc nélkül a felismerő rendszer nem működhet.

Arc detektálása után az arról készült kép elmentése kerül. Adott számú mentett arc után a döntéshozás algoritmusa lát munkához. A referencia és aktuálisan kivont mikroképek normalizált keresztkorrelációja kiszámításra kerül a következő formula alapján:

$$R(x,y) = \frac{\sum_{x',y'}(T(x',y') * I(x+x',y+y'))}{\sqrt{\sum_{x',y'} T(x',y')^2 + \sum_{x',y'} I(x+x',y+y')^2}}$$

ahol *T* a referencia kép, *I* pedig az aktuális, összehasonlítandó kép. Esetünkben egy korrelációs értéket kapunk az összehasonlítás eredményeként, mivel a képeink megegyező méretűek. Innen egy lokális döntés:

$$lokális\ döntés \begin{cases} igen & ha\ R(T,I) \geq küszöbszint \\ nem & ha\ R(T,I) < küszöbszint \\ ismeretlen & sikertelen\ jellemzőkinyerés \end{cases}$$

ahol *R(T,I)* a referencia és az összehasonlítandó kép keresztkorrelációjának értéke, a küszöbszint pedig a tanulási fázisból dinamikusan, autokorrelációkból származtatott érték. Innen egy arcra vonatkoztatott döntés:

$$arc\ szintű\ döntés \begin{cases} extra\ igen & ha\ i \geq e * n \\ igen & ha\ i > t * n \\ nem & ha\ i < t * n \\ extra\ nem & ha\ i < (1-e) * n \\ ismeretlen & ha\ u \geq (1-e) * n \end{cases}$$

ahol: i azon lokális döntések száma, amelyek eredménye „igen”, e egy magas érték a [0,1]-ből, t alacsony érték [0,1]-ből, $n=rf*nf$, nf a jellemzők száma, rf a jellemzőnként rögzített képek száma, u az „ismeretlen” lokális döntések száma. Jelenlegi megvalósításban: nf=5, rf=5, e=0,8, t=0,5. Innen a végleges döntés:

$$\text{végleges döntés} \begin{cases} igen & ha\ e_i \geq w * f \\ nem & ha\ e_n \geq w * f \\ ismeretlen & ha\ e_i = 1\ és\ n_i < t * (f-1) \\ igen & ha\ e_i = 1\ és\ n_i \geq t * (f-1) \\ ismeretlen & ha\ e_n = 1\ és\ n_n < t * (f-1) \\ nem & ha\ e_n = 1\ és\ n_n \geq t * (f-1) \\ igen & ha\ n_i \geq h * f \\ nem & ha\ n_n > (1-h) * f \\ ismeretlen & egyébként \end{cases}$$

ahol: f az arci képek (döntések) száma, e_i az „extra igen”, e_n az „extra nem”, n_i az „igen”, n_n a „nem” arconkénti döntések száma, w egy [0,1]-ből származó kis értékű szám, t ugyanaz, mint korábban, h egy [0,1]-ből származó szám. Jelenlegi megvalósításban h=t.

A döntéshozó „nem” kimenete esetén a rendszer egy e-mail illetve MMS üzenet küldését vonja maga után, amely tartalmaz egy képet az ismeretlen arcról. A riasztást követően ismét mozgásérzékeléssel folytatódik a felügyelet.

A döntéshozó „ismeretlen” kimenete esetén a rendszer hangjelzéssel kéri a megfigyelt területen tartózkodó személyt, hogy mondja be az azonosításhoz szükséges jelszót. Ennek ellenőrzésével teszi „igenné” vagy „nemmé” a bizonytalan kimenetet. Ha a jelszó nem megfelelő, a fentebb leírt riasztás indul el.

A döntéshozó „igen” kimenetére a rendszer nyugtázza a döntést és mozgásérzékeléssel folytatja munkáját. Egy elkészült döntést tartalmazó kép a 9. ábrán tekinthető meg (a képen a bal felső sarokban az éppen aktuális küszöbszintek arcrészek szerint feltüntetve, alatta a végleges döntés, alatta a lokális döntések, arconként oszlopba gyűjtve láthatók).

9. ábra: Döntéshozás eredménye

VI. A RENDSZER TESZTELÉSE

Az elkészült felügyelő rendszer tesztelése több módon és fázisban zajlott. Külön-külön teszteltük az egyes komponensek megbízhatóságát, majd az együttműködésük eredményét. Némely fázist több készüléken is vizsgáltunk. Ezek közt szerepelt: Pulid F11, 2x1GHz elérhető maximális felbontás 1280x720, ár: 20 000Ft (kínai); LG G3, 4x2,5GHz, használt felbontás 2048x1080, ár: 150 000Ft; Samsung S Advance, 2x1GHz, elérhető maximális felbontás 800x600, ár: 40 000Ft,

Az egyes komponensek vizsgálata során mindegyik gyors működést és nagyon jó eredményeket ért el. Magának az arcfelismerő algoritmusnak a tesztelése volt a másik fő feladat. Az ütemezett referencia készítésnek és betöltésnek a tesztelése sikeres volt. A személyek felismerését több módon is teszteltük. Az első tesztelési forma statikus állapotban, a referencia készítésének helyén történő mérés volt. A statikus állapot a személyre igaz, vagyis nem mozgott a képek készítése során, és a referencia készítés helyén és időpontjában történt a tesztelés. Ez a legjobb eshetőség, hiszen pontosan azokat az anomáliákat szünteti meg, amelyek a korrelációt megzavarják. A tesztek során a felhasználó szerepelt a képeken, tehát a válasz mindig „igen” kellett volna, hogy legyen. Az eredmények az 1-4. táblázatokban figyelhetők meg.

A nehezebb felismerési forma, amikor a személy nem vesz tudomást a készülékről és nem is néz bele, illetve mozgást végez, amely miatt a képek sem élesek. Nyilvánvaló, hogy itt nagyobb szerepet kap a hang alapú ellenőrzés. Maga a tesztelés egy megtanult mozgási sor végrehajtásával történt, hogy az eredmények összehasonlíthatóak legyenek. A 3. táblázatban szereplő mérések esetén a rendszer felhasználója, a megtanult személy felismerése volt a cél. A 4. táblázatban a mozgó, ismeretlen személy alapú mérések összegzése található.

A tesztelés eredményeit röviden összefoglalva elmondható, hogy az eszköz gyorsasága és minősége nem hat ki lényegesen az eredményekre. Az egyértelmű, hogy statikus helyzetben a rendszer megfelelően működik. Az idegen személyek azonosítása kicsivel gyengébb eredményeket hozott, de ha az illető nem ismeri a jelszót, akkor megfelelő a működés, hiszen a hang alapú azonosításé ilyen esetben a főszerep (de pontosan ezért is lett implementálva). Az eredményeket tekintve az elvárt működést tapasztaltuk, az ismert hiányosságok és korlátok jól látszódnak.

Eszköz	Mérések száma	Igen	Nem	Talán (megerősítve hang alapon)
Pulid F11	30	18	4	8 (6)
Samsung S Advance	20	10	3	7 (6)
LG G3	15	8	4	3 (0)

1. táblázat: Statikus, megtanult személy felismerése

Eszköz	Mérések száma	Igen	Nem	Talán (megerősítve hang alapon)
Pulid F11	20	3	10	7 (1)
Samsung S Advance	20	5	11	4 (0)
LG G3	0	0	0	0

2. táblázat: Statikus, ismeretlen személy felismerése.

Eszköz	Mérések száma	Igen	Nem	Talán (megerősítve hang alapon)
Pulid F11	20	5	6	9 (7)
Samsung S Advance	20	7	7	6 (6)
LG G3	10	2	4	4 (3)

3. táblázat: Dinamikus, ismert személy felismerése

Eszköz	Mérések száma	Igen	Nem	Talán (megerősítve hang alapon)
Pulid F11	20	1	3	16 (2)
Samsung S Advance	20	2	4	14 (0)
LG G3	10	2	3	5 (1)

4. táblázat: Dinamikus, nem ismert személy felismerése

VII. Összefoglalás

Kutatásunk célja egy olyan mobil alapú, intelligens, alacsony költségű felügyelő rendszer megalkothatóságának a vizsgálata és létrehozása volt, amely képes felismerni a felhasználója arcát. Sikerült egy olyan rendszert megalkotnunk, amely lehetővé teszi arcokról készült képek gyors gyűjtését, ezzel teret adva további, mélyebb kutatásoknak. A jelen megvalósításban szereplő felügyelő rendszer az arcról kivont mikrojellemzők (száj, orr, bal szem, jobb szem, szempár) képtérbeli összehasonlításával ítéli meg, hogy ismert arc szerepel-e egy adott képen. Habár sikerült így is megfelelő eredményeket elérnünk, nyilvánvaló, hogy ezen jellemzők nagyon érzékenyek a varianciákra. További fejlesztésként elsősorban ennek javítását tűzzük ki célul. A megvalósított algoritmusok közül különösen jól működött a hang alapú jelszóellenőrzés, amely érzékenységét a külső zajokra még javítani kell, de így is 85% feletti pontosságot ért el a tesztelés során.

A rendszer egyszerűen bővíthető, újabb jellemzők vizsgálata, bevétele könnyedén megoldható. A legtöbb eljárás paraméterezhető, tehát további kutatás folyamán is segítséget nyújt a szabad kísérletezésre. Az alkalmazott detektáló eljárások nagyon hatékonyan működnek, ezzel lehetővé téve az arcok gyors rögzítését. A saját kamera kezelő osztály további funkciók kifejlesztését teszi lehetővé, mint például az éjjeli üzemmód, a LED vaku ugyanis szabadon kapcsolható programból. Az alkalmazás Android alapokon íródott, így szinte tetszőleges eszközre változtatás nélkül feltelepíthető és azonnal használható. Az arcfelismerő funkció jelenleg statikus helyzetben lévő személyek esetén megfelelően biztonságosan, még mozgó egyének esetén, érthető korlátok miatt gyengébben szerepel. A tapasztalatok alapján úgy gondoljuk, hogy lehetséges egy olyan rendszer megalkotása, amely közel minden esetben képes egy adott arcról megfelelő döntést hozni. További bővítési és fejlesztési lehetőségek bőven adódnak: pontosított arcfelismerés, több megtanult arcra működő azonosítás, gépi tanulással támogatott döntéshozás, fejlettebb interakciós formák a felhasználóval, hogy csak néhányat említsünk. A rendszer tömegesebb alkalmazása esetén segítene felhasználni az egyébként kihasználatlan okostelefonokat és egyben segítene megóvni otthonainkat, értékeinket és támogatná a bűnüldözést.

References

1. OpenCV: www.opencv.org
2. N. Dalal and B. Triggs. Histogram of oriented gradients for human detection, Proc. IEEE Int. Conf. Comput. Vis. Pattern Recog., pp.886 -893 Jun. 2005
3. http://docs.opencv.org/modules/gpu/doc/object_detection.html
4. P. Viola and M. Jones. Rapid object detection using a bossted cascade of simple features. In Proc. IEEE Conference on Computer Vision and Pattern Recognition, 2001. 5. http://docs.opencv.org/modules/objdetect/doc/cascade_classification.html
6. Z. Zivkovic. Improvedadaptive Gaussian mixture model for background subtraction,Proc. Int. Conf. PatternRecognit., pp.28 -31 2004
7.http://docs.opencv.org/java/org/opencv/video/BackgroundSubtractorMOG2.html

A Google új, kísérleti QUIC protokolljának teljesítményelemzése

Krämer Zsolt, Megyesi Péter, Molnár Sándor

Nagysebességű Hálózatok Laboratóriuma,
Távközlési és Médiainformatikai Tanszék,
Budapesti Műszaki és Gazdaságtudományi Egyetem,
1117, Magyar tudósok körútja 2.,
Budapest, Magyarország
{kramer,megyesi,molnar}@tmit.bme.hu

***Kivonat*—A webes forgalom átvitelének meghatározó protokollja a 90-es évek óta a HTTP. Az elmúlt 20 évben azonban a weboldalak és webes alkalmazások olyannyira megváltoztak, hogy a protokoll működése a mai Interneten már nem optimális, teljesítményének korlátairól számos publikáció született. A Google az elmúlt években két új protokollt is implementált: 2009-ben a SPDY-t és 2013-ban a QUIC-et. A SPDY fejlesztése során világossá vált, hogy sok esetben a teljesítményt a szállítási rétegben használt TCP protokoll korlátozza. A kísérleti QUIC (Quick UDP Internet Connections) protokoll a hagyományokkal szakítva UDP fölött működik, és mivel a technológia még nagyon friss, ezidáig igen kevés kutatási eredmény került nyilvánosságra a működéséről. Írásunkban összefoglaljuk a SPDY és QUIC protokollok újításait, majd bemutatunk egy átfogó összehasonlító elemzést a QUIC, SPDY és HTTP protokollok teljesítményéről, a weboldalak letöltési idejére koncentrálva. Az eredmények alapján kijelenthető, hogy a legjobb teljesítményű protokollt a hálózat- és a weboldal paraméterei együtt határozzák meg, tehát nem létezik a három közül a körülményektől függetlenül leggyorsabb technológia.**

I. Bevezetés

A mai Internet egyik legszélesebb körben használt protokollja a HTTP (Hypertext Transfer Protocol), mely hírek, videók és megszámlálhatatlan webes alkalmazás átviteléért felel eszközök milliárdjain, az asztali számítógépektől az okostelefonokig. A weboldalaink azonban jelentősen megváltoztak a HTTP/1.1 [1] publikálása óta. Ahogy a webes alkalmazások tovább fejlődnek, az Internet forgalma pedig rohamosan nő, egyre nagyobb jelentőséggel bír az új technológiák kutatása. Az oldalletöltési idő (Page Load Time, PLT) különösen fontos aspektusa a teljesítménynek, melyre a weboldalak korai elhagyásával való kapcsolat [6] is rávilágít.

A Google 2009-ben kezdett bele egy új web transzfer protokoll fejlesztésébe. Ez a SPDY [2], melyet mára számos nagy forgalmú szerveren (Google, Facebook, Twitter) implementáltak, és az összes modern böngésző támogatja. A SPDY a szabványosítás alatt álló HTTP/2 protokoll alapja [4].

Azonban nem csak a HTTP/1.1-nek vannak hátrányai és korlátai. A szállítási rétegben a TCP (Transmission Control Protocol), mely a múltban kimagasló eredményekkel szolgált, szintén problémákkal küzd. A Google-nek lehetősége van hatékonyan elemezni a TCP teljesítményét a mai hálózatokban, mivel az Internet forgalmának 20-25%-a [7] az ő szerverein halad át, a Chrome böngésző pedig több, mint 45%-os piaci részesedéssel a piacvezető webböngésző [8]. Ezekre a tapasztalatokra építve fogott bele a Google a QUIC [3] fejlesztésébe 2013-ban. A QUIC protokoll a SPDY újításait egy új transzport protokoll fölé helyezve mind az alkalmazási-, mind a szállítási rétegben igyekszik áttörést elérni, és ezzel felgyorsítani a Webet.

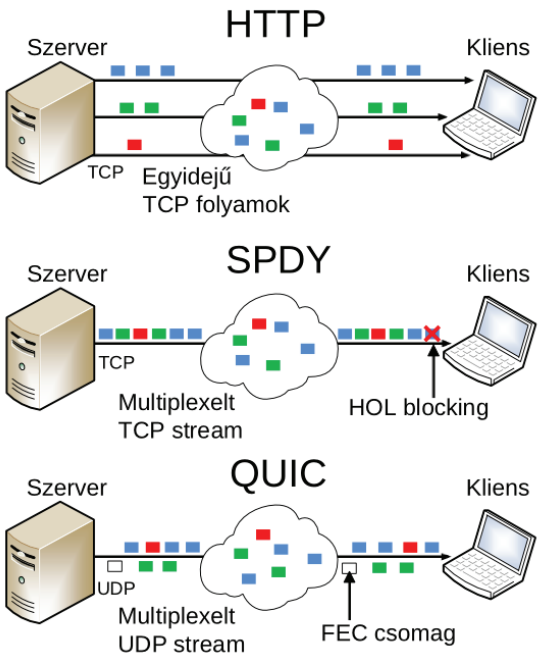
Mivel a QUIC még egy nagyon friss technológia, igen kevés tanulmány vizsgálta eddig. Írásunk elsődleges célja, hogy hozzájáruljon a QUIC protokoll teljesítményének alaposabb megértéséhez, összehasonlítva azt a HTTP/1.1-el és a SPDY-vel.

A II. fejezet összefoglalja a SPDY és a QUIC hátterét, illetve a kapcsolódó irodalmat. A III. fejezetben részletesen bemutatjuk a mérési környezetet, amit a protokollok teljesítményének teszteléséhez használtunk. A IV. fejezet tartalmazza a mérések eredményeit, majd az V. fejezetben összefoglaljuk a tapasztalatokat.

II. A Web protokolljainak evolúciója

II-A. A HTTP/1.1 hiányosságai

A HTTP/1.1 [1] egy kérés-válasz alapú protokoll, melyet a 90-es években terveztek, amikor a weboldalak még jelentősen egyszerűbbek voltak, mint ma. A felhasználói interakciókra ma már közel valós idejű választ várunk egy weboldaltól, melyet a HTTP/1.1 nem képes kiszolgálni. Az egyik legfontosabb, a HTTP teljesítményét hátrányosan befolyásoló mechanizmus a túl sok TCP kapcsolat nyitása a párhuzamosság elérése érdekében. A HTTP folyamok nagy része kis (15KB-nál kisebb), börsztös adatátvitelből áll, több tucat különböző TCP kapcsolaton, ahogy azt az 1. ábra szemlélteti. A TCP azonban hosszú kapcsolatokra optimalizált protokoll, az új kapcsolatok felépítésének költségét pedig nagyban befolyásolja a körülfordulási idő (Round-Trip Time, RTT) nagysága. Amikor a HTTP új TCP kapcsolatok nyitásával igyekszik javítani a teljesítményt, a túl nagy számú TCP kapcsolat torlódáshoz vezet, mely végeredményben rosszabb teljesítményt eredményez. A kapcsolatok száma még tovább nő, amennyiben a webes objektumok különböző domain-ekről érkeznek. A problémára a HTTP pipelining próbált megoldást kínálni, mely azonban nem terjedt el [5].



1. ábra. Adatfolyamok a HTTP, SPDY és QUIC protokollokban

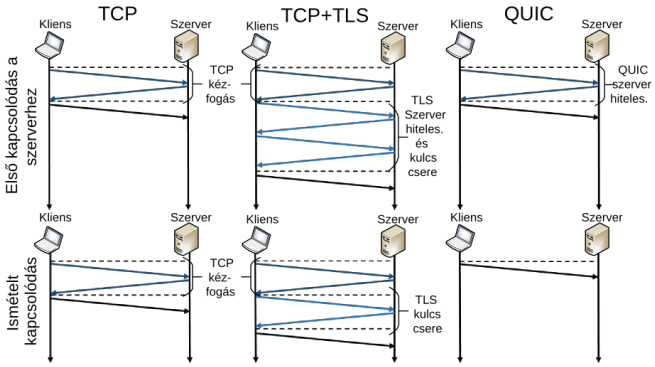
Szintén problémát jelent, hogy a HTTP-ben adatátvitelt kizárólag a kliens kezdeményezhet. Ez különösen beágyazott objektumok letöltésekor jár komoly teljesítménycsökkenéssel. A szervernek ilyenkor minden objektum küldése előtt várnia kell a kliens explicit kérésére, mely csak az után érkezhet meg, hogy a kliens feldolgozta a HTML dokumentumot.

Mivel egy TCP szegmens nem tartalmazhat egynél több HTTP kérést vagy választ, a kliensek jelentős mennyiségű redundáns adatot küldenek TCP SYN csomagok és HTTP fejlécek formájában. Ez a feleslegesen átvitt adatmennyiség különösen naggyá válik a hasznos adathoz képest, amikor sok kis méretű beágyazott objektum található az oldalon. ADSL (Asymmetric Digital Subscriber Line) kapcsolatok esetén, ahol a feltöltési sávszélesség különösen szűkös erőforrás, a redundáns adatok átvitele komolyan megnövelheti a késleltetést. A fejlesztői közösség a múltban az azonos típusú, kis méretű fájlok konkatenálásával igyekezett ezt kivédeni, illetve néhány esetben inline fájlok használatával a HTML dokumentumban. Ezek az eljárások azonban komoly hátrányokkal is rendelkeztek: a fájlok konkatenálása negatív hatással van a cache hatékonyságára, illetve a CSS és JavaScript fájlok konkatenálása késlelteti a feldolgozást [5].

II-B. SPDY

A SPDY[1] protokoll tervezésekor a cél az volt, hogy orvosolja a HTTP említett hiányosságait [2]. A protokoll az alkalmazási rétegben működik, TCP fölött. A SPDY keretező rétege a HTTP-hez hasonlóan kérés-válasz streamekre optimalizált, így azok az alkalmazások, amik HTTP fölött futottak, SPDY

[1]A SPDY “speedy”-ként ejtendő és nem egy mozaikszó



2. ábra. A kapcsolat felépítése a TCP, TLS és QUIC protokollokban

fölött is futhatnak, akár változtatások nélkül. A továbbiakban bemutatjuk a SPDY főbb újításait.

Ahogy az 1. ábrán is látható, a SPDY egy domain-hez egyetlen, multiplexelt TCP kapcsolatot nyit. Az egy kapcsolat (SPDY session) fölött, párhuzamosan kiszolgált kérések száma tetszőlegesen nagy lehet. Ezek a kérések stream-eket, kétirányú adatfolyamokat hoznak létre. Ez a multiplexelés a HTTP pipelining-nál jóval kifinomultabb eljárás a kérések párhuzamosságának biztosítására, mert csökkenti az SSL (Secure Sockets Layer) overhead-et, növeli a szerverek hatékonyságát és segít a torlódáselkerülésben. A stream-ek létrejöhetnek a kliens- vagy a szerveroldalon, és szállíthatnak más streamekkel átlapolt adatot [2].

A SPDY újításai közé tartozik a kérések priorizálhatósága. A kliensnek lehetősége van egy prioritási szintet rendelni minden objektumhoz, majd a szerver ezen prioritásoknak megfelelően ütemezi az objektumok átvitelét. Ez segít elkerülni az olyan eseteket, amikor a csatornán nem kritikus objektumok torlódást okoznak, míg egy magas prioritású kérés (pl egy JavaScript kód modul) várakozni kényszerül.

A protokollban a szerver a kliens explicit kérése nélkül is küldhet adatot a kliensnek (szerver oldali push-mechanizmus). Enélkül a kliensnek először le kell töltenie a HTML dokumentumot és csak ezután kezdeményezheti a másodlagos erőforrások átvitelét. A szerver oldali push-mechanizmus csökkentheti a késleltetést beágyazott objektumok letöltésekor, azonban ronthat a cache hatékonyságán, így a megfelelő optimalizáció kulcsfontosságú.

A SPDY a HTTP fejlécek formájában átvitt redundáns adatmennyiség problémájára is kínál megoldást: a fejléceket egy dedikált stream-ben, tömörített formában viszi át. Egészen a SPDY 3-as verziójáig a protokoll a DEFLATE formátumot használta a redundáns adatok leképezésére, azonban ezzel az eljárással kapcsolatban súlyos sebezhetőségek kerültek napvilágra, mint például a CRIME (Compression Ratio Info-leak Made Easy). Erre válaszul a SPDY 4-es verziójában már a HPACK eljárással kódolják a HTTP fejléceket [9].

A SPDY teljesítményét a mai napig nem értjük megfelelően, ezt bizonyítja, hogy a témában egymásnak ellentmondó kutatási eredmények láttak napvilágot. A Google [12]

és a Microsoft [13] jelentős teljesítményjavulásról számolt be (60% csökkenés a PLT-ben) a HTTP-vel összehasonlítva, ezzel ellentétben az Akamai [14] és a Cable Labs [15] eredményei csupán alacsony mértékű javulást mutatnak, sőt, néhány esetben teljesítményromlást. [16] és [17] szintén kis mértékű teljesítményjavulást mutattak ki nagy körülfordulási idő mellett (például műholdas kapcsolatok esetén), azonban egy másik tanulmány [18] kimutatta, hogy ez 3G hálózatokban nem érvényesül a cellás rendszerek felépítése miatt.

[10] és [11] izolált teszthálózatban vizsgálta a SPDY teljesítményét, nagy kiterjedésű paramétertérben. A két publikáció hasonló eredményeket tartalmaz: i) a SPDY jobban teljesít a HTTP-nél nagy számú objektum vagy nagy RTT esetén, legfőkébb a multiplexelésnek és a tömörített fejléceknek köszönhetően, ii) a SPDY nagyobb teljesítményjavulást ér el alacsony sávszélesség esetén, nagy sávszélesség mellett az eredmények között nincs számottevő különbség, iii) a HTTP jobban teljesít a SPDY-nél nagy csomagvesztés mellett, ahol a SPDY teljesítménye drámaian visszaesik az ú.n. head-of-line blocking (HOL blocking) miatt (bővebben lásd a következő alfejezetet). Ezeket az eredményeket a mi méréseink is megerősítették.

II-C. A TCP korlátai

A SPDY sikeresen túllépett a HTTP/1.1 számos hiányosságán, azonban akadnak a további teljesítményjavulást akadályozó tényezők, melyeket a szállítási rétegben kell keresnünk. A TCP egyik legfontosabb funkciója a torlódásszabályozás. Az évek során számos új TCP verziót javasoltak (például [19], [20]), illetve egyes kutatócsoportok új, alternatív transzport protokollokat is fejlesztettek (például [21]). Sajnos azonban a szállítási réteg protokolljaival, elsősorban a TCP-vel kapcsolatban a tapasztalatok azt mutatják, hogy a protokoll nagyon lassan fejlődik, a fejlesztések pedig még ennél is lassabban terjednek el. Ennek oka, hogy nem csupán szervereken és klienseken kell implementálni, hanem middlebox-ok tömegén szerte az Interneten. Ez azt jelenti, hogy egy újítás a TCP protokollban 10 év-, vagy akár ennél is hosszabb idő alatt terjed el széles körben.

A sávszélesség ma már egyre kevésbé korlátozza a webes teljesítményt (többek közt a lakossági üvegszálas megoldások elterjedése miatt), azonban a késleltetésről gyakran megfeledkezünk. A hálózati RTT a legfontosabb faktor egy új TCP kapcsolat throughput-jában, és értékét nagyban behatárolja a fény terjedési sebessége, így a körülfordulások számának csökkentése az egyetlen út, amivel a késleltetés jelentősen csökkenthető. Ahogy a 2. ábrán látható, a TCP-nek egy körülfordulásra van szüksége, hogy felépítse a kapcsolatot mielőtt a HTTP kérés elküldhető válik. Ha titkosított kapcsolatról beszélünk, akkor ez az idő tovább növekszik; legalább egy RTT-re van szükség a TLS kulcs-cseréhez, az első kapcsolódás esetén pedig még egy körülfordulásra az azonosításhoz.

Fontos még megemlítenünk a head-of-line blocking jelenségét. A TCP megbízható kapcsolatot és sorrendhelyes átvitelt garantál, ez pedig azt jelenti, hogy ha egy csomag elveszik, minden adatküldésnek várnia kell az újraküldésig. [10] és [11] megmutatták, hogy magas csomagvesztés mellett a HTTP jobban teljesít a SPDY-nél, mert a SPDY esetén egy domainhez egyetlen TCP kapcsolat létezik, és a HOL blocking ez esetben az összes stream-et várakozásra kényszeríti, míg a HTTP a több párhuzamosan létező TCP kapcsolat miatt kevésbé érintett.

II-D. QUIC

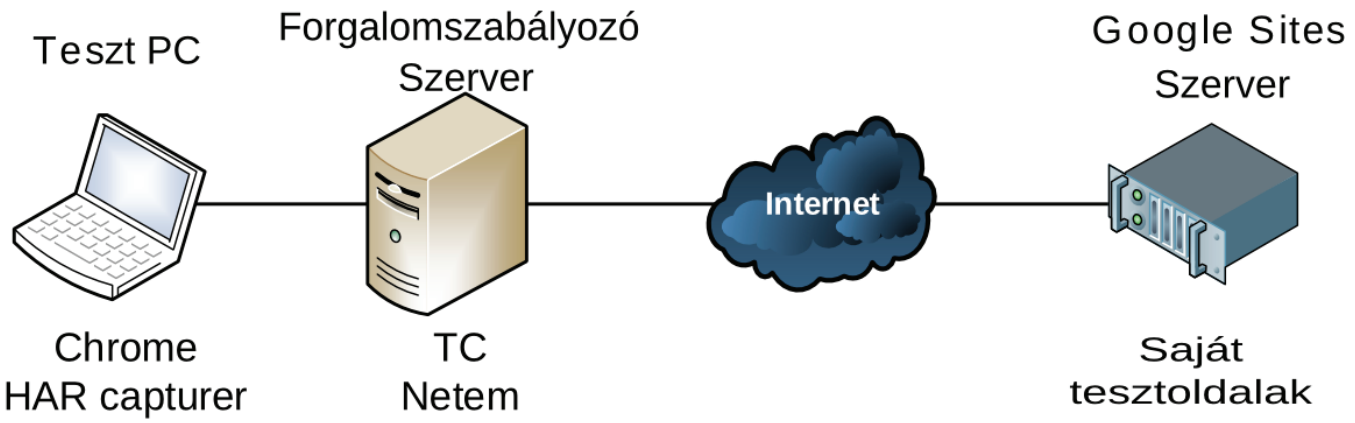
A QUIC protokoll [3] fejlesztésekor az egyik legfontosabb célkitűzés a webes forgalom késleltetésének csökkentése volt. Ehhez szükség volt áttérni TCP-ről UDP-re a transzport rétegben, illetve egy új titkosító protokoll implementálására, mely kiváltja a TLS/SSL-t, így támogatva a kapcsolat felépítéséhez szükséges körülfordulások számának csökkentését, miközben a biztonsága a TLS-sel összemérhető marad [3].

Mivel a QUIC UDP fölött működik, nincs garancia a csomagok sorrendhelyes átvitelére, így elkerülhető a HOL blocking. Egy kliens akár 0-RTT alatt csatlakozhat egy szerverhez, ha korábban már létezett kapcsolat közöttük (lásd a 2. ábrán). Ez úgy érhető el, hogy minden csomag tartalmaz egy, a kapcsolatra vonatkozó azonosítót, mely kiváltja a hagyományos IP fourtuple-t (forrás és cél címek, illetve portok). A plusz körülfordulás a TLS-ben nem követelmény a biztonság szempontjából, kizárólag a kézfogás implementációjából származik. A QUIC-ben implementált titkosító ezen változtat, működése pedig a DTLS-hez (Datagram Transport Layer Security) hasonló [22].

Ahogy az 1. ábrán látható, a QUIC protokoll hibajavító kódolás (FEC) használatával is igyekszik kiküszöbölni a csomagvesztés negatív hatását a teljesítményre. A QUIC hajlandó áldozni a hasznos sávszélességből a késleltetés csökkentéséért a kritikus csomagok (mint például a kapcsolatot kezdeményező UDP csomag) proaktív újraküldésével. A Google mérései alapján a FEC csomagokra fordított 5% extra sávszélesség 8%-kal kevesebb újraküldést eredményez [23].

A torlódásszabályozás megvalósítása a QUIC protokollban logikailag a TCP-CUBIC-kal (illetve opcionálisan a TCP-Reno-val) megegyezően van implementálva. Ezt azonban kiegészíti a packet pacing nevű mechanizmus, mely folyamatos optimalizálás alatt áll. A packet pacing vezérléséhez a QUIC a csomagküldési idők közötti különbségből becsüli meg a rendelkezésre álló sávszélességet, és szabályozza a csomagküldés sebességét. Korai mérések azt mutatták, hogy a packet pacing csökkenti a torlódásból származó csomagvesztések számát [23], azonban a teljesítményt jelentősen ronthatja alacsony csomagvesztési arány mellett [24].

A Google a korai eredményeit [22]-ben és [23]-ben mutatta be, illetve [24] is vizsgálta a QUIC teljesítményét, azonban ezekben az esetekben a metodológia leírása teljesen hiányzik. [25] megállapította, hogy a QUIC magas csomagvesztés esetén jobban teljesít a SPDY-nél, és hogy a FEC modul aktiválása lassítja a QUIC-et. [26] eredményei alapján a QUIC nagy RTT és alacsony sávszélesség mellett képes jobban teljesíteni a SPDY-nél és a HTTP-nél. Az utóbbi két kutatásban a Google publikus QUIC prototípus-szerverét [28] használták, mely a méréseket maximálisan megismételhetővé teszi, ám az



3. ábra. Az összeállított mérési elrendezés

eredmények pontosságát mégis veszélyezteti (bővebben lásd a III. fejezetet).

III. Mérési környezet

Ebben a fejezetben részletesen ismertetjük a metodológiát, aminek segítségével a QUIC, SPDY és HTTP protokollok teljesítményét összehasonlítottuk. A 3. ábra szemlélteti az alkalmazott tesztkörnyezetet. Egy laptopon[2] futtatott Chrome böngészőn a Chrome HAR Capturer [27] segítségével automatizáltuk az oldalletöltéseket, az eredményeket pedig HAR(HTTP ARchive) formátumban mentettük el. A HAR fájlok a letöltési idők vizsgálatához szükséges minden információt tartalmaznak. A Chrome HAR Capturer minden egyes oldalletöltés előtt törli a socket pool-t és a cache-t.

Készítettünk 4 különböző weboldalt, melyeket a Google szerverein (Google Sites) helyeztünk el. Azért ezt a megközelítést választottuk, mert QUIC szervereket eddig kizárólag a Google használ (pl Gmail, YouTube, Google Translate), ezeken kívül pedig csak egy egyszerű szerver modul érhető el, melyen az implementáció tesztelhető [28], azonban a teljesítmény korrekt elemzésére és összehasonlítására nem feltétlenül alkalmas. A méréseinkhez használt egyetemi hálózat és a Google Sites szerver között nagyon alacsony ingadozást tapasztaltunk a sávszélességben és a körülfordulási időben, emiatt úgy véljük, hogy az élő szerveren végzett mérések megfelelő pontosságú eredményekkel szolgálhattak.

A QUIC és a SPDY is multiplexelt kapcsolatot használ, melynek előnye elsősorban olyan weboldalakon jelentkezik, amin nagy számú objektum található. A Google Sites-on elhelyezett weboldalaink kis- (400B-8KB) vagy nagy (128KB) méretű, illetve kis (5) vagy nagy (50) számú objektumot tartalmaztak. Az objektumok képek: a kis méretűek nemzeti zászlók, a nagy méretűek pedig nagy felbontású fényképek[3].

A különböző hálózati körülményeket a netem [30] (része a Linux Traffic Control csomagjának) használatával emuláltuk, így állítottuk be az egyes hálózati konfigurációkat meghatározó

[2]Intel Core i5 CPU, 4GB RAM, Ubuntu 14.04 (64bit), Chrome verzió:37.0.2062.94.

[3]A Google Sites automatikusan átméretezi a nagy méretű fotókat 128KB-ra.

Kategória	Paraméter	Értékek
Hálózati	Sávszélesség	2 Mbps, 10 Mbps, 50 Mbps
	RTT	18 ms, 218 ms
	Csomagvesztés	0%, 2%
Weboldal	Objektumok száma	5, 50
	Objektumok mérete	400 B - 8 KB, 128 KB

I. táblázat. Az egyes szcenáriókat meghatározó paraméterek

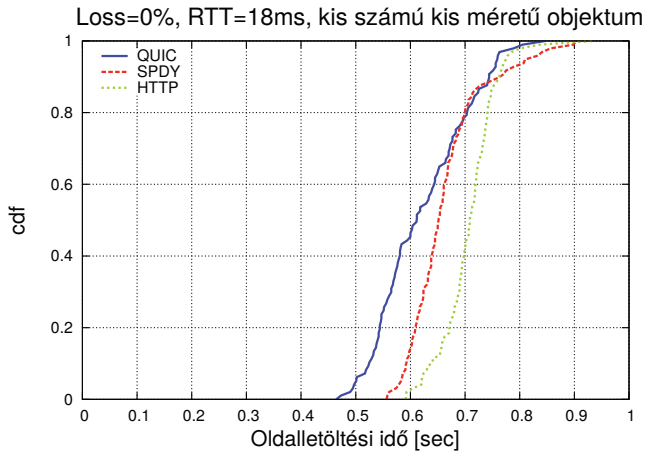
sávszélesség, csomagvesztés és késleltetés értékeket. Sávszélességben az alacsony, közepes és magas értékeket 2 Mbps, 10 Mbps, illetve 50 Mbps-ként definiáltuk. A csomagvesztés hatását kétféle beállítás mellett vizsgáltuk: alacsony-, amikor nem adunk a hálózathoz extra csomagvesztést, illetve magas, amikor a TC segítségével mind kimenő-, mind a bejövő csomagok 2%-át véletlenszerűen eldobjuk. A késleltetés esetében az alacsony beállításnál nem adtunk a hálózathoz extra késleltetést, így az átlagos RTT 18 ms volt, míg magas késleltetésnél mindkét irányban 100 ms-os késleltetést emuláltunk, így az átlagos RTT 218 ms lett.

Az I. táblázatban látható az öt dimenziós paramétertér összefoglalása. Ez 48 különböző konfigurációt jelent, ahol minden esetben legalább 100 mérést végeztünk.

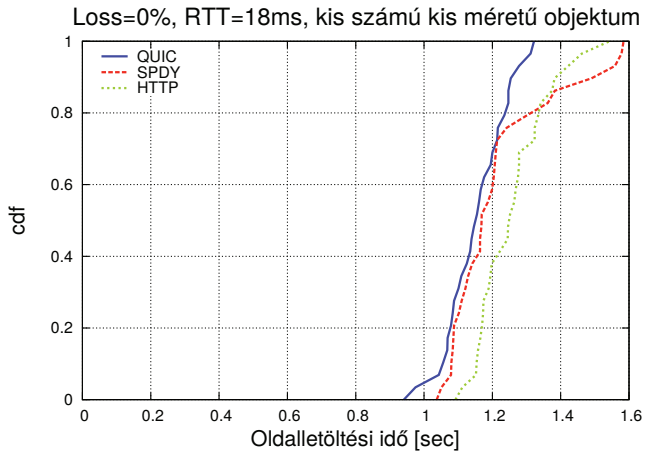
IV. Eredmények

Ebben a fejezetben összehasonlítjuk, hogy a QUIC, SPDY és HTTP protokollok milyen gyorsan tudják letölteni a Google Sites szerveren elhelyezett különböző weboldalakat. A cikk terjedelmi korlátai miatt nem mutatjuk be mind a 48 különböző szcenáriót, ehelyett néhány kiválasztott konfiguráció eredményeire fókuszálunk, melyek érzékeltethetik a főbb tanulságokat.

A 4. ábrán látható az oldalletöltési idő (PLT) eloszlásfüggvénye (CDF) alacsony csomagvesztés(loss) és RTT mellett a legkisebb oldalon (kis számú kis méretű objektum). 4a mutatja az 50 Mbps -, 4b pedig a 10 Mbps sávszélességhez tartozó eredményeket. Mindkét esetben csak kis különbség látszik a protokollok teljesítményében. Az átlagok ugyan némileg

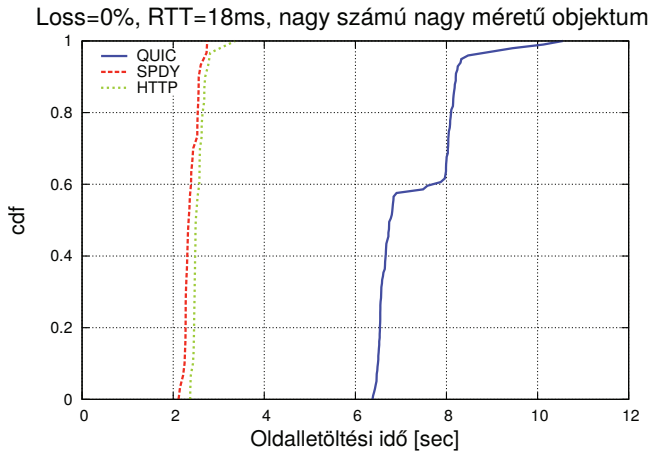


(a) Bw = 50Mbps

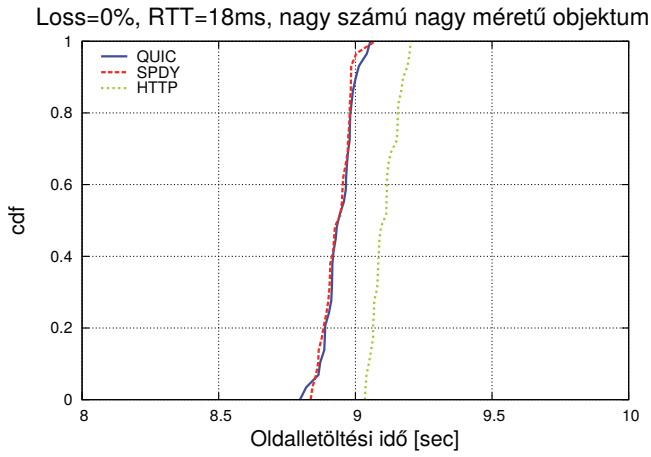


(b) Bw = 10Mbps

4. ábra. Az oldalletöltési idők eloszlása alacsony RTT, alacsony csomagvesztés és kis számú kis méretű objektum esetén



(a) Bw = 50Mbps



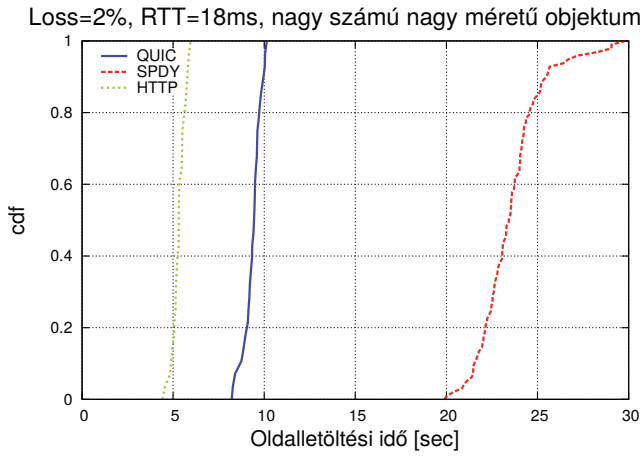
(b) Bw = 10Mbps

5. ábra. Az oldalletöltési idők eloszlása alacsony RTT, alacsony csomagvesztés és nagy számú nagy méretű objektum esetén

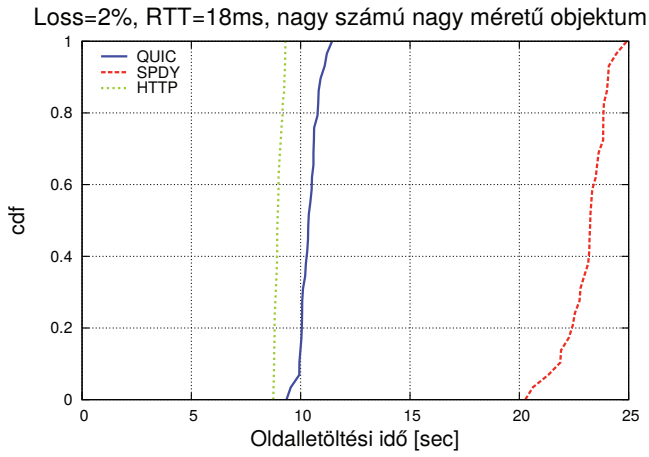
különböznek, de a görbék szórása nagyobb ennél a különbségnél, így nem jelenthetjük ki, hogy bármelyik protokoll egyértelműen gyorsabb lenne a másik kettőnél. Az események nagyon hasonlóak voltak a nagy számú kis méretű objektumot tartalmazó -, és a kis számú nagy méretű objektumot tartalmazó weboldalon. Megállapíthatjuk tehát, hogy a protokollok nagyjából egyformán teljesítenek jó hálózati körülmények között (nagy sávszélesség, alacsony csomagvesztés, alacsony RTT) kis-, vagy közepes méretű oldalak letöltésekor.

Jóval nagyobb különbség mutatkozik a protokollok teljesítményében nagy méretű weboldalak esetén. A 5. ábra alacsony csomagvesztés és RTT mellett ábrázolja az értékeket a nagy számú nagy méretű objektumot tartalmazó oldal letöltésekor. Amikor a sávszélességet 10 Mbps-ra állítottuk (5b), a letöltési idők ismét hasonlóak lettek, azonban az 50 Mbps-os szcenárióban (5a) a QUIC jelentősen rosszabbul teljesít a SPDY-nél és a HTTP-nél. Ebben az esetben a QUIC átlagos oldalletöltési ideje több, mint háromszorosa a másik két protokollénak. Ennek fő oka a QUIC packet pacing mechanizmusa: a QUIC goodput-ja nem képes elérni egy ilyen link kapacitását. Mivel ez a viselkedés sem a 10 Mbps-os konfigurációnál, sem a kisebb méretű oldalak letöltésekor nem jelentkezett, arra következtethetünk, hogy a QUIC teljesítménye csak nagy rendelkezésre álló sávszélesség, és nagy letöltendő adatmennyiség mellett csökken drasztikusan a másik két protokollhoz képest.

A 6. ábra és a 5. ábra konfigurációja kizárólag a magas csomagvesztésben különbözik. Ebben az esetben a SPDY nagyon rosszul teljesít: az átlagos PLT többszörösére nőtt a 2%-os csomagvesztés hozzáadása után. A HOL blocking drámai negatív hatását a SPDY teljesítményére korábbi írások is bemutatták, mint [10], [11], ekkora teljesítménycsökkenést azonban egyik kutatás sem mutatott ki, mert a szerzők nem vizsgálták a csomagvesztés hatását ilyen magas sávszélesség mellett. Az eredményekből az is látszik, hogy QUIC-et használva az átlagos PLT az alacsony csomagvesztésű esethez képest csak 20%-kal nőtt, míg a HTTP-nél nagyjából megduplázódott. Ez elsősorban a QUIC gyorsabb csomagvesztés utáni helyreállási mechanizmusaival magyarázható, másodsorban pedig a FEC használatával. A 7. ábrán alacsony sávszélesség és magas RTT

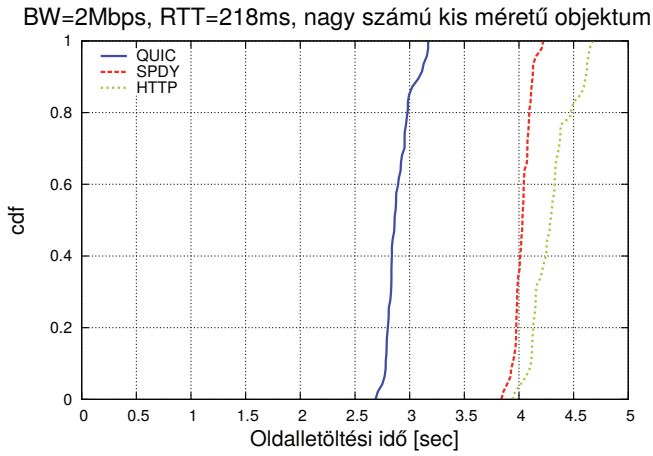


(a) Bw = 50Mbps

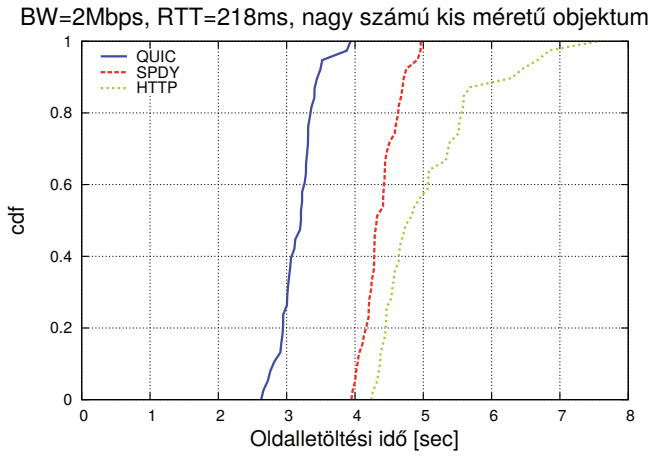


(b) Bw = 10Mbps

6. ábra. Az oldalletöltési idők eloszlása alacsony RTT, magas csomagvesztés és nagy számú kis méretű objektum esetén



(a) Alacsony csomagvesztés



(b) 2% csomagvesztés

7. ábra. Az oldalletöltési idők eloszlása 2 Mbps sávszélesség, magas RTT és nagy számú kis méretű objektum esetén

mellett láthatóak a nagy számú kis méretű objektumot tartalmazó oldal letöltési idői veszteségmentes (7a) és veszteséges (7b) környezetben. Az eredményeink igazolják a feltételezést, miszerint a SPDY és QUIC multiplexelt kapcsolatai komoly előnyt jelentenek sok kis méretű objektum letöltése esetén. A QUIC 0-RTT kapcsolódási ideje is komoly jelentőséggel bírt: a QUIC messze a leggyorsabb protokoll, 25-30%-kal megelőzve a SPDY-t és 35-40%-kal a HTTP-t. Fontos még rámutatni arra, hogy a SPDY és a HTTP teljesítménye között a különbség 7-12%, ami megerősíti azokat az állításokat a SPDY irodalmában, melyek szerint a protokoll csak kis mértékben gyorsabb a HTTP-nél.

A mérési eredményeinket a 8. ábrán foglaljuk össze, egy döntési fa formájában. Egy protokollt akkor tekintünk jobbnak egy másiknál egy adott szcenárióban, ha az esetek legalább 70%-ában legalább 10%-kal alacsonyabb PLT-vel rendelkezik. Amennyiben két protokoll gyorsabbnak bizonyult a harmadiknál, de egymásnál nem, a fa megfelelő levelén mindkettőt feltüntettük. Azokban az esetekben, ahol sem a leggyorsabb, sem a leglassabb protokollt nem lehetett a fenti kritérium alapján meghatározni, a levelet egyenlő eredményként jelöltük meg. A döntési fa és a teljes mérési adatbázis alapján az alábbi következtetéseket vonhatjuk le:

-A QUIC rosszul teljesít, amikor magas sávszélesség mellett nagy mennyiségű adatot kell letölteni

-Másfelől, a QUIC remekül teljesít a másik két protokollal összehasonlítva magas RTT mellett, különösen, ha a sávszélesség alacsony

-A magas csomagvesztés kevésbé rontja a QUIC teljesítményét, mint a másik két protokollét

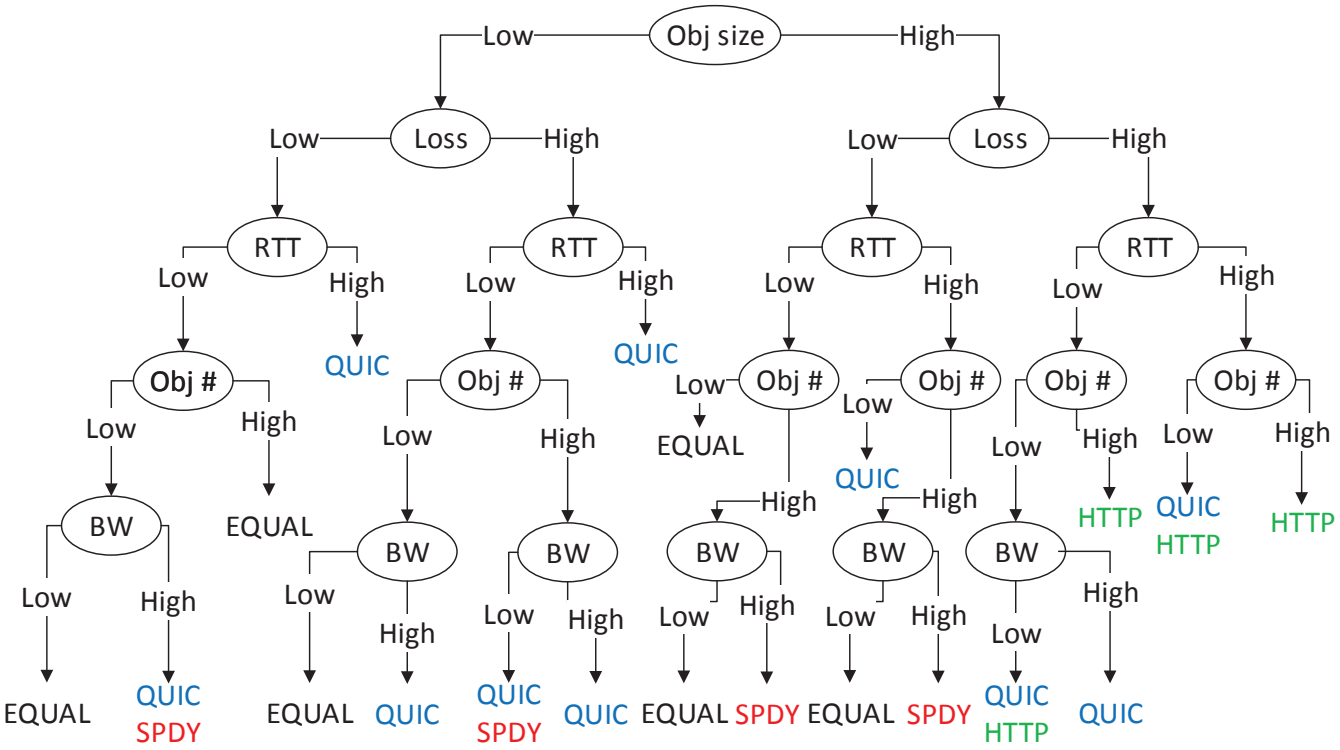
-A SPDY teljesítménye nagyon érzékeny a csomagvesztésre a HOL blocking miatt

-A kis méretű objektumok átvitelekor a multiplexelt kapcsolat előnyt jelent

-Nagy sávszélesség, magas csomagvesztés, és nagy számú nagy méretű objektum esetén a HTTP a leggyorsabb protokoll

V. Konklúzió

A QUIC (Quick UDP Internet Connections) egy új protokoll, melyet a Google 2013 óta fejleszt, célja pedig a SPDY-



8. ábra. Döntési fa, mely hozzárendeli a leggyorsabb protokollt a paramétertér adott pontjához

hez képest további nyereség elérése a szállítási rétegben, ezzel a webes forgalom gyorsabb átvitele. A TCP használata helyett a protokollt UDP fölé implementálták. Ebben a tanulmányban bemutattunk egy összehasonlító elemzést a QUIC, SPDY és HTTP protokollok teljesítményéről, és beazonosítottuk az ezt leginkább meghatározó környezeti paramétereket. Építettünk egy egyszerű tesztkörnyezetet, melyben egy laptopot használva töltöttünk le különböző weboldalakat a Google Sites szerverről, és egy shaper szerver használatával különböző hálózati körülményeket emuláltunk.

Az esetek több, mint 40%-ában a kísérleti QUIC protokoll jelentősen javított a letöltési időkön, de az eredményeink azt is megmutatják, hogy a HTTP képes jobban teljesíteni a multiplexelt protokolloknál nagyméretű objektumok letöltésekor. Az eredmények alátámasztják a korábbi publikációk ([10], [11]) állítását, miszerint a SPDY teljesítményére nagyon erős negatív hatással van a csomagvesztés.

Nagy RTT értékek mellett a QUIC kiemelkedően jól teljesített. Mivel a mobil Internet kapcsolatok (különösen a 3G) esetén a késleltetés általában magas, amikor a QUIC protokoll kilép a kísérleti állapotból, javasolt az implementálása azokon a webszervereken, melyek nagy mobil forgalmat bonyolítanak. A protokoll szintén alapértelmezetten használható lehet a Chrome böngészőben Android platformon.

Az egyetlen beazonosított körülmény, mely a QUIC teljesítményét negatívan befolyásolta a másik két protokollhoz képest, a nagy sávszélesség volt. Ennek egyik magyarázata a packet pacing mechanizmus lehet, mely megakadályozza, hogy a protokoll kihasználhassa egy nagysebességű link kapacitását. Szintén problémát okozhat, ha a hálózatüzemeltetők biztonsági megfontolásokból korlátozzák az UDP forgalmat [31]. A QUIC fölötti webes forgalom, és a veszélyesnek ítélt UDP forgalmak hatékony megkülönböztetésének kidolgozása a fejlesztők és a hálózatüzemeltetők közös feladata a közeljövőben.

Terveink között szerepel a vizsgálatok folytatása, és a QUIC protokoll fejlődésének nyomon követése.

KÖSZÖNETNYILVÁNÍTÁS

A szerzők szeretnék megköszönni Szabó Géza és Rácz Sándor (TrafficLab, Ericsson Research Hungary) ötleteit és hasznos megjegyzéseit a cikkel kapcsolatban.

HIVATKOZÁSOK

[1] Hypertext Transfer Protocol (HTTP/1.1) - RFC 2616. Letöltve: 2015.09.04. Elérhető online: http://tools.ietf.org/html/rfc2616

[2] SPDY Protocol - Draft 3. Letöltve: 2015.09.04. Elérhető online: http://www.chromium.org/spdy/spdy-protocol/spdy-protocol-draft3

[3] QUIC: A UDP-Based Secure and Reliable Transport for HTTP/2. Letöltve: 2015.09.04. Elérhető online: http://tools.ietf.org/html/draft-tsvwg-quic-protocol-01

[4] Hypertext Transfer Protocol Version 2 (HTTP/2) - RFC 7540. Letöltve: 2015.09.04. Elérhető online: https://tools.ietf.org/html/rfc7540

[5] Grigorik, I.: Making the web faster with HTTP 2.0. Communications of the ACM, Vol. 56, No. 12, pp. 42–49 (2013)

[6] Diriye, A., White, R., Buscher, G., Dumais, S.: Leaving so soon?: understanding and predicting web search abandonment rationales. In Proceedings of the 21st ACM International Conference on Information and Knowledge Management (CIKM '12), New York, NY, USA (2012)

[7] Sandvine - Global Internet Phenomena. Letöltve: 2015.09.04. Elérhető online: https://www.sandvine.com/trends/global-internet-phenomena/
[8] Market Share of Web Browsers (September 2014). Letöltve: 2015.09.04. Elérhető online: http://www.w3counter.com/globalstats.php?year=2015&month=8
[9] HPACK: Header Compression for HTTP/2 - RFC 7541. Letöltve: 2015.09.04. Elérhető online: https://tools.ietf.org/html/rfc7541
[10] Wang, X. S., Balasubramanian, A., Krishnamurthy, A., Wetherall, D.: How Speedy is SPDY? In Proceedings of the 11th USENIX Conference on Networked Systems Design and Implementation, Seattle, WA, USA (2014)
[11] Elkhatib, Y., Tyson, G., Welzl M.: Can SPDY Really Make the Web Faster? In Proceedings of IFIP Networking Conference, Trondheim, Norway (2014)
[12] SPDY: An experimental protocol for a faster web. White Paper. Letöltve: 2015.09.04. Elérhető online: http://www.chromium.org/spdy/spdy-whitepaper
[13] Padhyeand, J., Nielsen, H. F.: A Comparison of SPDY and HTTP Performance. Microsoft Technical Repor MSR-TR-2012-102.
[14] Podjarny, G.: Not as SPDY as You Thought. Letöltve: 2015.09.04. Elérhető online: http://www.guypo.com/technical/not-as-spdy-as-you-thought/
[15] White, G., Mule, J. F., Rice, D.: Analysis of SPDY and TCP Initcwnd. White Paper. Letöltve: 2015.09.04. Elérhető online: http://tools.ietf.org/html/draft-white-httpbis-spdy-analysis-00
[16] Cardaci, A., Caviglione, L., Gotta, A., Tonellotto, N.: Performance Evaluation of SPDY Over High Latency Satellite Channels. In Personal Satellite Services, Lecture Notes of the Institute for Computer Sciences, Social Informatics and Telecommunications Engineering, Vol. 123, pp. 123–134 (2013)
[17] Wang, X. S., Balasubramanian, A., Krishnamurthy, A., Wetherall D.: Demystifying page load performance with WProf. In Proceedings of the 10th USENIX Symposium on Networked Systems Design and Implementation (NSDI '13), Lombard, IL, USA (2013)
[18] Erman, J., Gopalakrishnan, V., Jana, R., Ramakrishnan, K. K.:Towards a SPDY'ier mobile web? In Proceedings of the 9th International Conference on emerging Networking EXperiments and Technologies (CoNEXT'13), Santa Barbara, CA, USA (2013)
[19] Afanasyev, A., Tilley, N., Reiher, P., Kleinrock, L.: Host-to-Host Congestion Control for TCP. In IEEE Communications Surveys and Tutorials, Vol. 12, No. 3, pp. 304–342 (2010)
[20] Molnár, S., Sonkoly, B., Trinh, T. A.: A Comprehensive TCP Fairness Analysis in High Speed Networks. Computer Communications, Elsevier, Vol. 32, No. 13-14, pp. 1460–1484 (2009)
[21] Molnár, S., Móczár, Z., Sonkoly, B.: How to Transfer Flows Efficiently via the Internet? In Proceedings of International Conference on Computing, Networking and Communications (ICNC 2014), Honolulu, USA (2014)
[22] QUIC: Design Document and Specification Rationale. Letöltve: 2015.09.04. Elérhető online: https://docs.google.com/document/d/1RNHkx_VvKWyWg6Lr8SZ-saqsQx7rFV-ev2jRFUoVD34/edit
[23] Roskind, J.: QUIC - Multiplexed Stream Transport over UDP. IETF-88 TSV Area Presentation. Letöltve: 2015.09.04. Elérhető online: http://www.ietf.org/proceedings/88/slides/slides-88-tsvarea-10.pdf
[24] Taking Google's QUIC For a Test Drive. Letöltve: 2015.09.04. Elérhető online: http://www.connectify.me/taking-google-quic-for-a-test-drive/
[25] Carlucci, G., De Cicco, L., Mascolo, S.: HTTP over UDP: an Experimental Investigation of QUIC. In Proceedings of The 30th ACM/SIGAPP Symposium On Applied Computing (SAC 2015), Salamanca, Spain (2015)
[26] Das, R. S.: Evaluation of QUIC on Web Page Performance. Master's Thesis, Massachusetts Institute of Technology, MA, USA, 2014.
[27] Chrome HAR Capturer. Letöltve: 2015.09.04. Elérhető online: https://github.com/cyrus-and/chrome-har-capturer
[28] QUIC Test Server. Letöltve: 2015.09.04. Elérhető online: http://www.chromium.org/quic/playing-with-quic
[29] Apache SPDY Module. Letöltve: 2015.09.04. Elérhető online: https://code.google.com/p/mod-spdy/
[30] Hemminger, S.: Network Emulation with NetEm. In Proceedings of the 6th Australia's National Linux Conference (LCA2005), Canberra, Australia (2005)
[31] Byrne, C.: Advisory Guidelines for UDP Deployment. Letöltve: 2015.09.04. Elérhető online: https://tools.ietf.org/html/draft-byrne-opsec-udp-advisory-00

Integrált tömegfelügyeleti rendszer okos városokban

Nagy Attila Mátyás

MSc hallgató, Budapesti Műszaki és Gazdaság Tudományi Egyetem, Hálózati Rendszerek és Szolgáltatások Tanszék

***Absztrakt*— A tömegrendezvények óriási látogatottságnak örvendenek, ugyanakkor számtalan veszélyt is hordoznak magukban. Nagy tömegekben már egy kisebb pánik — amelyek megjóslása, megelőzése komoly feladat — is beláthatatlan következményekkel járhat, amellett, hogy a szervezők egyébként mindent megtesznek a résztvevők biztonságáért.**

A mi megoldásunk egy közösségi érzékelésen alapuló integrált tömegfelügyeleti rendszer, amelynek segítségével a biztonságért felelős szervek, közel valós időben, átfogó képet kaphatnak a tömeg helyzetéről, mozgásáról. Az új információk alapján gyorsan és célzottan lehet beavatkozni, így akár életeket is mentve.

***Kulcsszavak*— tömegfelügyelet, mobil érzékelés, tömeg érzékelés, okos városok**

I. BEVEZETÉS

HOGYAN tudnánk megelőzni egy tömegpánik kialakulását? Esetleg hogyan tudjuk észrevenni, hogy a tömeg már akkorára duzzadt, hogy nemes egyszerűséggel összenyomnák egymást az emberek? Hogyan lehetne előrelátni az ilyen vészhelyzetek kialakulását? Gyakran előállhatnak hasonló váratlan helyzetek, melyek a rendfenntartó szervektől és a szervezőktől is a lehető leggyorsabb reakciót kívánják. Minden egyes alkalommal más-más biztonsági kérdéssel kell megbirkózniuk és ekkora létszámok esetén ez nem kis feladat és felelősség. Ebből adódóan, sajnos minden évben előfordulnak tömegkatasztrófák, melyeknek egy része megelőzhető lenne, ha a szervezők számára több információ állna rendelkezésre, bár vannak olyan előre nem megjósolható események, amelyekre nagyon nehéz felkészülni.

2006-ban Mekkában a Jamarat hídnál 345 ember vesztette életét, amikor a túl nagy tömegben eltaposták egymást. Erről az esetről készült egy tanulmány is, ahol videó felvételek alapján elemzik a tömeg mozgását. Egy másik ismertebb, a média által felkapott eset a Love Parade volt 2010-ben Németországban, ahol 21 ember halt meg és 500-an megsérültek. Mindkettő elkerülhető lett volna megfelelő beengedés-szabályozás használatával. Valószínűleg, ahogy növekeszik a világ népessége, fejlődik a tömegközlekedés, egyre jobban éreztetik hatásukat a globalizációs folyamatok, egyre gyakoribbá válhatnak ezek a szomorú esetek, emiatt a szervezőket is fel kell vértezni olyan támogató rendszerekkel, szoftverekkel, amelyek segítségével jobban megfigyelhető, megérthető a tömegek mozgása, és akár előreláthatók és elkerülhetők lennének a katasztrófák.

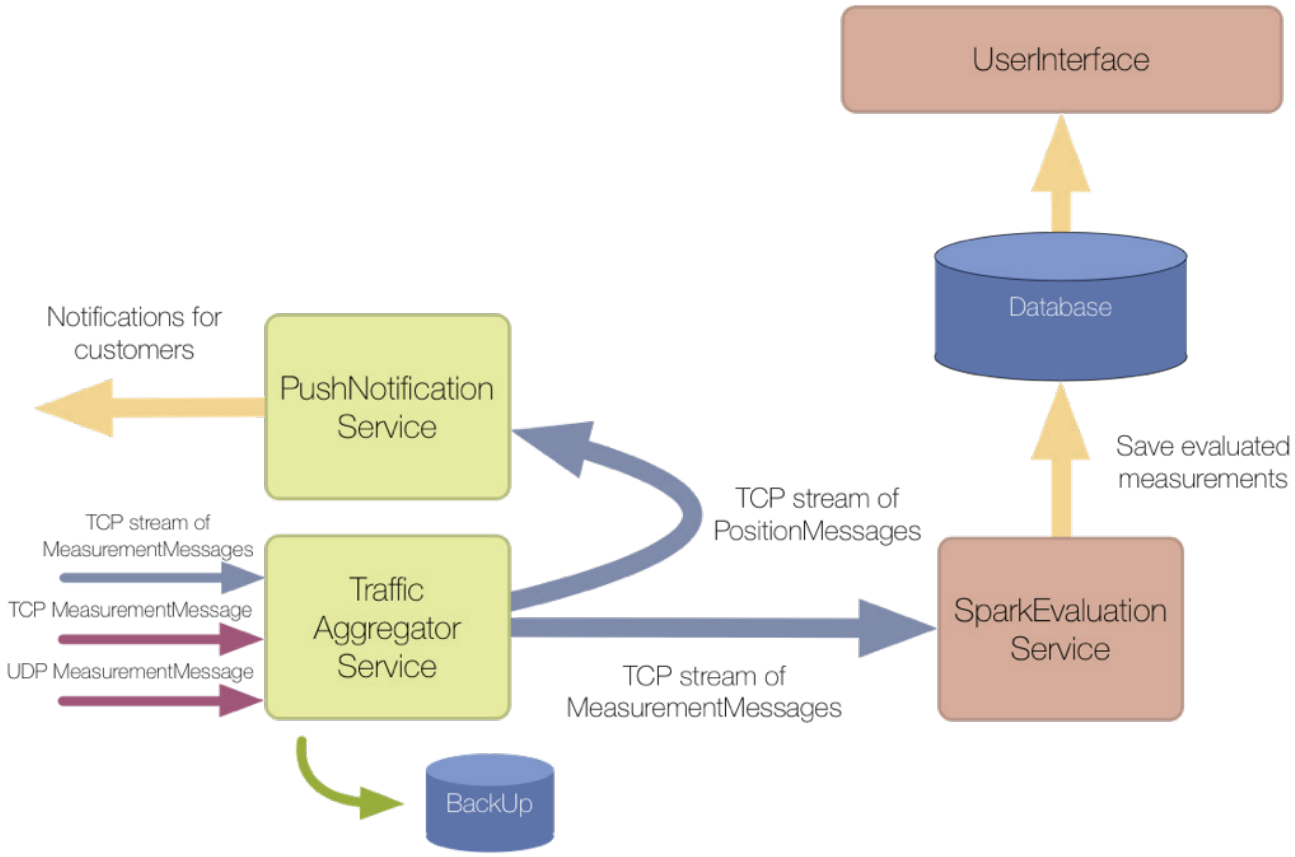
A mai tömegrendezvények látogatóinak túlnyomó többsége már rendelkezik olyan mobil készülékkel (okostelefonnal, tablettel), amelyekben különféle szenzorok (GPS, giroszkóp) találhatóak meg. A szenzorokból származó adatok összegyűjthetőek és ezekből rengeteg információ kinyerhető, így lehetőségünk van monitorozni a tömeg dinamikáját, mozgását, akár becsléseket készíteni a jövőbeli állapotokról. Ez több szempontból is nagyon hasznos lehet. Egy nagy rendezvény esetén, bár vannak elképzeléseink és pontatlan becsléseink a tömeg eloszlásáról, nem tudjuk pontosan, hogy egy területen mennyi ember tartózkodik egy adott pillanatban. Ha ezt mérni tudnánk, a tömeg eloszlásától függően képesek lehetnénk a rendfenntartó egységek helyesebb átcsoportosítására és dinamikusan módosíthatnánk az evakuációs terveket, ha a szükség úgy kívánja. A megoldás továbbá lehetővé teszi, hogy a résztvevők számára, pozíciójuk alapján különböző üzeneteket küldjünk. Ezek az üzenetek lehetnek közérdekű felhívások, de akár reklámüzenetek is.

II. KAPCSOLÓDÓ MUNKÁK

Léteznek már olyan rendszerek, amelyek különböző felügyeleti feladatokat látnak el, viszont ezek nem feltétlenül a bevezetésben megfogalmazott feladatokkal foglalkoznak, vagy másképpen oldják meg a problémát.

Manapság a kamera alapú felügyeleti rendszerek a legelterjedtebbek a tömegfelügyeleti rendszerek esetében. Egy 2011 márciusában született cikk [1] szerint az Egyesült Királyság területén eddig 1,85 millió kamerát telepítettek, bár ezek közül 1,7 millió privát rendszerekben teljesít szolgálatot. Rendkívül sok helyzetben alkalmazzák őket, természetesen nem csak rendezvények megfigyelésére. A városi gyalogos forgalom vagy jármű forgalom vizsgálatával, tanulmányozásával hatékonyabbá tehetjük a létező rendszereket, folyamokat. Autópályák esetén a kamerarendszer segítségével egyből észrevehetőek a balesetek, hamarabb lehet értesíteni a mentőket, rendőröket, a baleset mögött haladó forgalom egy részét el lehet terelni, így csökkenthető a dugók mérete, kialakulásának valószínűsége, de egy átlagos napon is a kamerák adatainak feldolgozásával hasznos információkat lehet közölni a vezetőkkel. Meghatározó szerepe lehet még a bűnmegelőzésben és bűnüldözésben is. A kamerák segítségével a megfigyelt tömegben felfedezhetőek a körözött személyek, vagy egy bűntényről készült videó felvétel kulcsfontosságú információkat tartalmazhat az ügy megoldásához vagy a bíróságon terhelő bizonyíték lehet a tettes ellen.

A szakirodalomban felfedezhető bluetooth alapú megoldás is. A rendszer [2] fejlesztésénél a cél az volt, hogy egy olyan



1. ábra: A tömegfelügyeleti rendszerarchitektúrája. Az ábrán láthatóak a főbb komponensek, illetve hogy ezek milyen típusú üzenetekkel kommunikálnak.

mobil alkalmazást készítsenek, ami segítséget nyújt a menedzsmentben és a kommunikációban a szervezőknek egy nagyméretű szabadtéri eseményen. Ezeken a rendezvényeken a kritikus információk az elsősegély pontokhoz, az emberek irányításához vagy a mobilitáshoz (parkolók, ajánlott útvonalak) kapcsolódnak. Fő céljaik a következőek voltak:

- Tömeg méretének, sűrűségének, mozgásának mérése automatizmusok segítségével;
- A mért adatok transzformálása a szervezők számára hasznos információvá;
- Biztosítani egy dedikált kommunikációs csatornát a hatóságoknak és a szervezőknek, hogy értesíteni tudják az eseményen résztvevőket;
- Megbizonyosodni, hogy a kommunikációs kapcsolat robosztus és hibavédett; hogy elérhető legyen vészhelyzet esetén is, amikor a celluláris kommunikáció esetleg csődöt mond.

A tömeg méretének, eloszlásának érzékeléséhez a Traffax által fejlesztett szenzorokat használják [3]. A berendezéseket eredetileg jármű forgalom méréséhez tervezték, de alkalmas gyalogos forgalom vizsgálatára is.
Egy másik publikált rendszer [4] teljesen más értelmezésben és méretekben közelíti meg a felügyeleti rendszer fogalmát. Míg az előző egyértelműen egy tömegrendezvény biztonságosabbá tételével foglalkozott, addig ez a rendszer nem egy ekkora méretű eseményre koncentrál, hanem arra, hogy megoldható-e, hogy egy nagyobb területet, például egész megyéket, országokat figyeljünk meg adott pillanatban és detektálhatunk-e különböző vészhelyzeteket.
Az elmúlt években a közösségi oldalak igen népszerű médiummá váltak gyakorlatilag a világ majdnem minden táján, emiatt rendkívül jó információforrás és gyors kommunikációs csatorna lehet, különösen természeti katasztrófák esetén. Az egyik ilyen szolgáltatás a Twitter, melynek segítségével a felhasználók rövid szöveges üzeneteket (maximum 140 karakteres tweeteket) oszthatnak meg követőikkel webes és mobil alapú platformok használatával. A Twitter egyik legfontosabb jellemzője a valósidejűsége. A felhasználók gyakran osztanak meg információkat arról, hogy mi jár a fejükben, mit látnak, mit tapasztalnak. A fejlett országokban a legtöbb város területén redundáns a hálózat, elérhető egyszerre vezetékes, WiMax, Wi-Fi, vagy celluláris hálózat, amelyek közül mindegyiken lehet a szolgáltatás segítségével kommunikálni. Ha valahol baj történne, és erről valaki írna egy tweetet (például hogy mit lát, mi történt), az a biztonsági szervezetek számára igen releváns információkat tartalmazna.
Természetesen nem lenne megoldás, ha innentől kezdve egy csapat éjjel-nappal figyelné az adott területről beérkezett

Twitter üzeneteket és keresné a biztonsági szempontból hasznos tweeteket, hanem szükség van egy olyan rendszerre, ami megbirkózik a közösségi média által szolgáltatott óriási adatfolyammal, és ebből különböző adatbányászati technikák segítségével kiválasztja a lényeges üzeneteket. Ezek a technikák a burst detekció, szöveg klasszifikáció online klaszterezés, vagy a geotaggelés.

Az általunk tervezett rendszerre legjobban hasonlító megoldás egy 2013-ban publikált cikkben [5] található. A rendszer egy általános keretrendszert — amit egy adott eseményre testre lehet szabni — alkalmaz arra, hogy a résztvevőktől másodpercenként GPS pozíciókat gyűjtsenek, amelyeket később a mobilszervereknek továbbít, ahol azokat ki is értékelik. Az adatok feldolgozása után hőtérképet állítanak elő a különböző tömegjellemzők szemléltetésére:

- sűrűség
- sebesség
- turbulencia
- tömegnyomás [6]

A hőtérképen a meleg színek jelölik az egyes jellemzők magas értékét, a színezés átlátszósága pedig az ottani sűrűséget mutatja. Ezáltal jól felismerhetők a sűrű és nagy sebességű / turbulenciájú / nyomású területek, amelyek a problémákat okozzák.
A cikkben röviden kitérnek a minták reprezentativitására is. Azt feltételezi, hogy az összes felhasználó statisztikailag a tömeggel arányosan oszlik el a területen, így helyes következtetések vonhatók le a tömeg viselkedéséről. Persze kívánatos minél több aktív felhasználó jelenléte és a mérések a CCTV felvételekkel pontosíthatók.
Megemlíti még, hogy az alkamazást a 2011-ben a Londoni Lord Mayor's Show-n is tesztelték, ahol a teszt előtt csak CCTV-vel monitorozták a tömeget. A rendfenntartók úgy találták, hogy a hőtérképes ábrázolás sokkal könnyebben adott globális képet a tömegről, a jellemzők sokkal könnyebben leolvashatók voltak. A résztvevőket is megkérdezték arról, hogy vészhelyzet esetén mennyire hagyatkoznának az alkalmazás által küldött tippekre. A válaszok alapján ez függne a vészhelyzet típusától, attól, hogy lenne-e a helyszínen hivatalos személy, a mobilon érkező információ hivatalos szervektől jön-e, az információ koherens-e a helyszínen tapasztaltakkal.
A cikk két fontos következtetést is levon. Első, hogy minél nagyobb felhasználószámra van szükség, hogy pontosabb becsléseket tudjon adni, tehát szükség van arra, hogy öszönözzék az emebreket arra, hogy feltelepítsék az alkalmazást a készülékeikre. Második fontos következtetés, hogy a felhasználókra szondaként kell tekinteni és akkor is lehetséges a pontos tömegjellemzők meghatározása, ha nem tudunk mindenkit követni.

III. TÖMEGFELÜGYELETI RENDSZER FEJLESZTÉSE

A.Követelmény a rendszerrel szemben

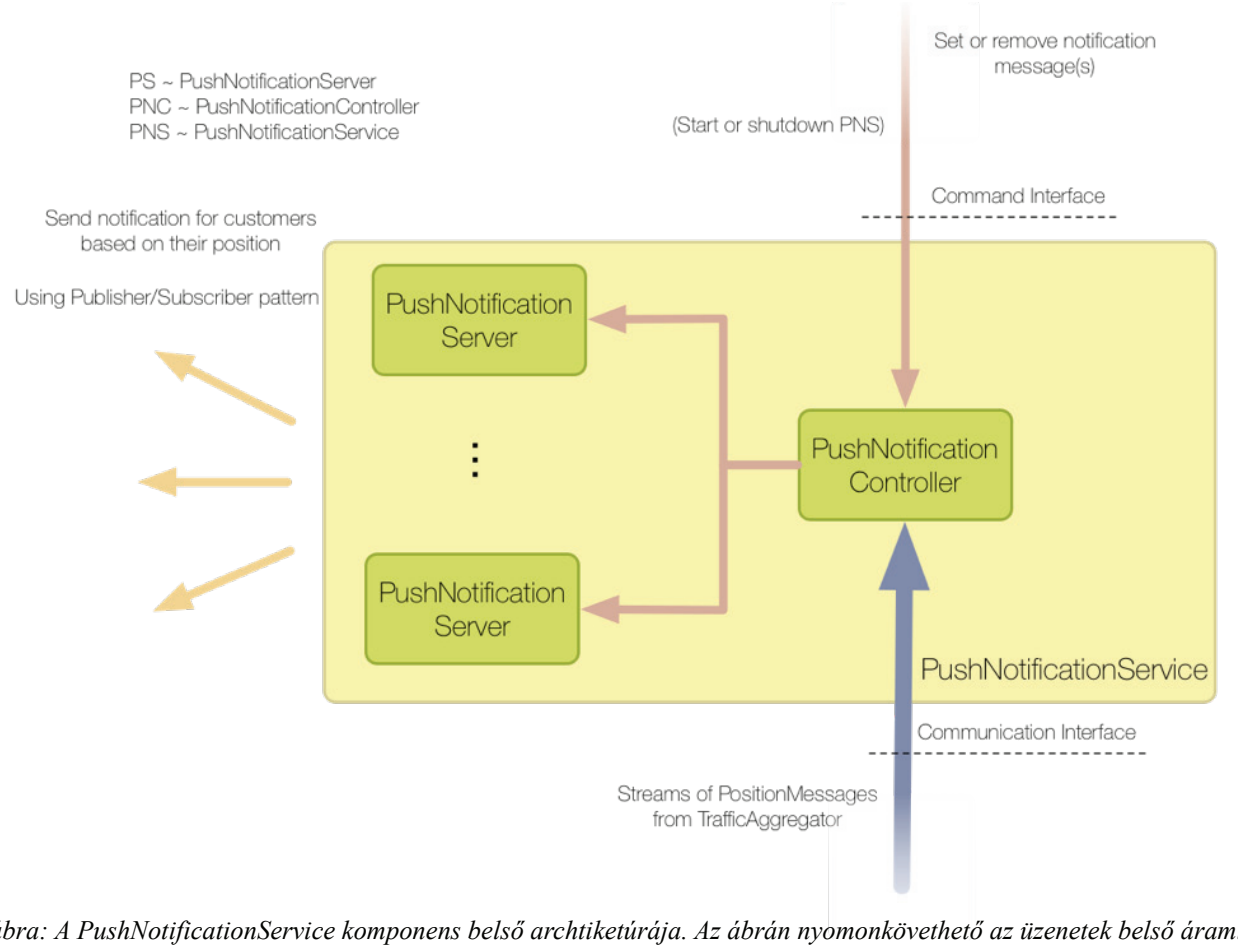
Az általunk fejlesztett rendszer is, ahogyan azt a bevezetésben említettük a mobil érzékelésen alapul. Abban az esetben, ha az applikáció népszerűvé válik fel kell készülni nagy adatmennyiség beérkezésére. Ez egyrészt azt is jelenti, hogy nagyszámú párhuzamos kapcsolat kezelésére kell képesnek lennie a rendszernek, hiszen könnyen előfordulhat, hogy egy száz ezres tömegből egy időpillanatban több ezren vagy akár tízezren is küldenek be adatokat, másrészt pedig olyan rendszerre van szükség, ami képes hatékonyan feldolgozni ezt a nagy mennyiségű mérést, akár szerver fürtökön elosztva, mivel az eredményekre közel valós időben van szükség. Az előzőek miatt nem kifizetődő folyamatosan adatokat gyűjteni minden egyes felhasználótól, mert ez kezelhetetlen méretű adathalmazt hozna létre, rengeteg szükségtelen állapotot is tárolna az rendszer, a készülékek akkumulátora nagyon hamar lemerülne és elképesztő méretű hálózati forgalmat is generálna. Jobb ötletnek tűnik, ha kevesebb adatot gyűjtünk, viszont azok releváns információkat tartalmaznak.

B.Tömegfelügyeleti modell

A követelményeket megfontolva alakítottunk ki egy olyan tömegfelügyeleti modellt, amely amellett, hogy alkalmas megfelelően ábrázolni az aktuális állapotot megfelelő absztrakciót is biztosít, melynek segítségével jelentősen lecsökkenthető az irreleváns adatok száma és így a hálózati forgalom is.
Első lépésben a teret diszkrét területekre bontottuk (cellás felbontás [7]). Fontos, hogy itt a cellák nem négyzetek, hanem tetszőleges konvex négyszögek. Ez azért fontos, mert a cellákat érdemesebb úgy elhelyezni, hogy a valamilyen szempontból összetartozó területeket ne bontsuk szét. Például nem célszerű úgy elhelyezni egy cellát, hogy a területének a fele egy forgalmas úton van, a másik pedig egy épületre lóg rá, ahol nyilván nem lesz senki. Ez pontatlanságot okozna a sűrűségek kiszámításánál, amely egyenesen arányos a nem kihasznált területtel. A sűrűség számításhoz még szükség van arra, hogy minden cellához megadjuk, hogy mekkora a területük.
Második lépésben a résztvevőket kellett elhelyezni a modellben. A gyalogos (vagy eszköz, hiszen jelen esetben ugyanazzal a mozgás állapottal rendelkeznek) aktuális állapotát két változó segítségével lehet jellemezni: a személy pozíciójával, sebességének ingadozásával. A pozíció esetén felesleges pontos GPS koordinátákat eltárolni, hiszen a sűrűségeket a kiértékelésénél úgyis cellákra határozzuk meg, emiatt elegendő, hogyha az eszköz csak cella azonosítókat küld el, melyik cellából melyikbe lépett. A sebesség ingadozás pedig ahhoz szükséges hogy a Lord Mayor's Show-n használt rendszerhez [5] hasonlóan mi is tudjunk tömegnyomást számolni, amely egy másik megfelelő metrika tömegvizsgálatára.

C.Rendszer architektúra

A rendszeren belül több különálló komponens működik együtt. Az egységek között a feladatok úgy lettek szétosztva, hogy egy komponens meghibásodása ne okozza a rendszer a teljes leállását. Az 1. ábrán nem jelenik meg, de a mobil eszközök egy applikáció segítségével tartják fent a kapcsolatot a szerverrel. Magát az applikációt nem mi tervezzük, hiszen egy rendezvény manapság már rendelkezik ilyennel, mi csak egy plugint biztosítunk, amit csak egyszerűen hozzá kell csatolni a már létező alkalmazáshoz. A plugin egy API-n keresztül hívható és amellett, hogy ki-be kapcsolható fel lehet iratkozni a szervertől érkező üzenetekre, amiket aztán már megjeleníthetnek a felhasználóknak.



2. ábra: A PushNotificationService komponens belső archtiketúrája. Az ábrán nyomonkövethető az üzenetek belső áramlása, és az elemek egymáshoz kapcsolódása.

TrafficAggregatorService

A rendszer egyik központi eleme a TrafficAggregatorService (továbbiakban TAS). Feladata, hogy felületet biztosítson a mobil eszközök számára, ahová beküldhetik pozíció és sebesség ingadozás méréseiket (MeasurementMessage). Ezután a beérkezett adatokat egy TCP folyamban továbbítja a kiértékelő komponens felé (SparkEvaluationService). Egy időpillanatban több kiértékelő szerver is csatlakozhat, így lehetővé téve, hogy akár egy egész szerver klaszter képes legyen résztvenni az adatok feldolgozásában.

Ezek alapján a TAS csak forgalom továbbításra szolgál és felmerül a kérdés, hogy egyáltalán miért van rá szükség. A későbbiekben részletesen kitérünk rá, de a kiértékelő komponens egy kliens-szerver architektúrában csak kliensként képes működni, emiatt önmagában nem képes fogadni a több tízezer forrásból érkező adatot. Ez teszi szükségessé a TAS működését.

Hogy képes legyen bírni a nagy terhelést, a mobil eszközöknek biztosított interfészén, a Netty aszinkron esemény vezérelt keretrendszerét [8] használjuk, melynek segítségével akár több tízezer párhuzamos kapcsolatot is lehet kezelni. Fontos megjegyezni, hogy az eszközök a minél hatékonyabb erőforrás kihasználás céljából nincsenek folytonos kapcsolatban a TAS-sal, hanem csak akkor küldenek adatot (mindössze 29 bájtban), hogyha az szükséges (cella váltáskor). Mivel kicsik az elküldött adatcsomagok, az esetek túlnyomó többségében egy TCP kapcsolat felépítése csak felesleges plusz kommunikációt jelentene, emiatt egyszerre biztosít TCP és UDP portokat is a kommunikációhoz. A mobil eszközökön lévő alkalmazások általános esetben az üzenetek az UDP portra, a vészhelyzetek esetén a TCP portra küldik, így biztosítva, hogy a fontos üzenetek biztosan ne vesszenek el.

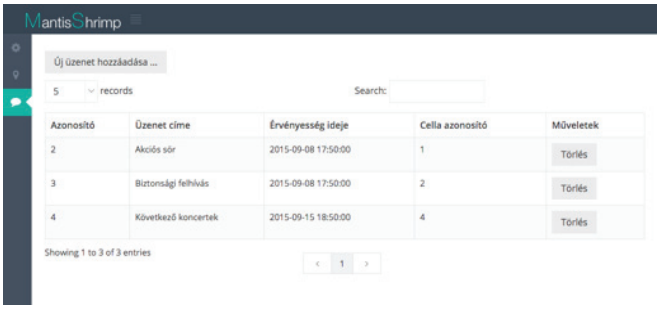
Abban az esetben, ha nem lenne elegendő egy TAS működése, lehetőség van arra is, hogy többet egymáshoz láncoljunk master-slave architektúrában, így növelve még jobban a teherbírást.

SparkEvaluationService

Ez a komponens felel a beérkező mérési adatok hatékony kiértékeléséért. Ehhez a Spark általános célú adatfeldolgozó motort [9] használjuk, amely képes elosztottan szerver klaszteren nagymennyiségű adatok kezelésére is. A Spark egy másik nagy előnye, hogy képes TCP streamből érkező adatok feldolgozására (SparkStreaming) is.

A komponens minden indulásánál, az adatbázisban szereplő adatok alapján felépíti az aktuális tömegállapotot. Erre azért van szükség, mert egy esetleges meghibásodás esetén is onnan tudja folytatni a kiértékelést, ahol abbahagyta.

A kiértékelő komponens a SparkStreaming segítségével elkezdi eltölteni a méréseket a TAS-tól. A letöltött adatokra



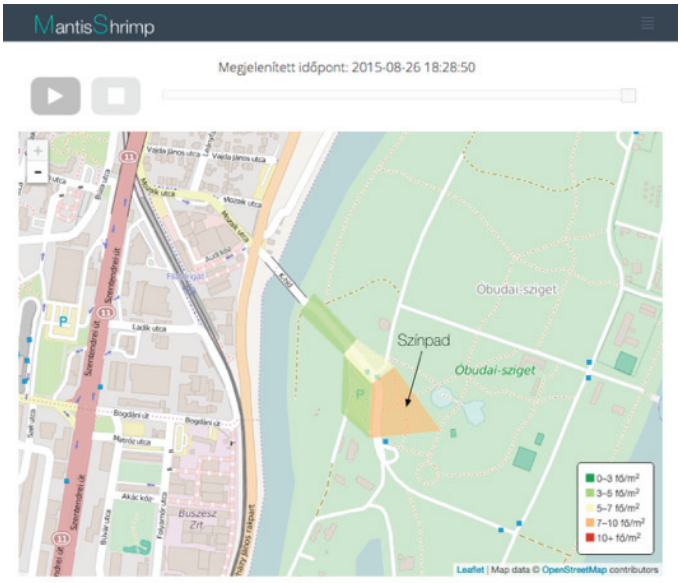
4. ábra: A PushNotificationService kezelő felülete. Ezen a felületen keresztül lehet üzeneteket küldeni a résztvevőknek.

ezután meghatározott időpontokban (például minden 30. másodpercben) lefuttatja a kiértékelő algoritmust. Az így kinyert információkkal először frissíti a saját belső állapotát, majd ezt kimenti az adatbázisba. Amúgy ez azaz adatbázis, amelyet a felhasználói felületen keresztül el lehet érni.

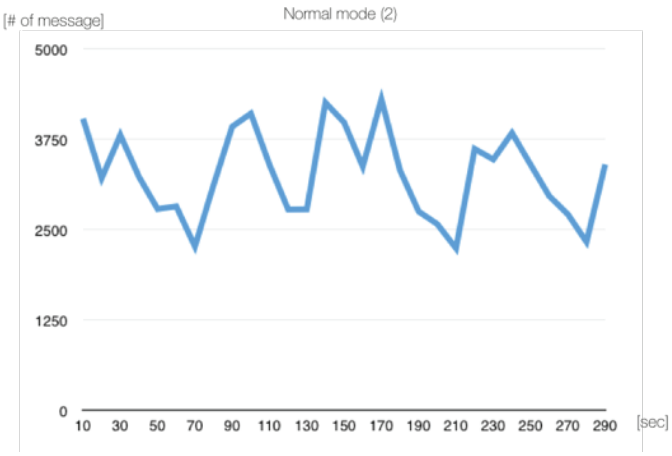
PushNotificationService

A PushNotificationService komponensnek (továbbiakban PNS) a feladata, hogy egy egyirányú kommunikációs csatornát biztosítson a rendfenntartók és a résztvevők között push jelleggel a pozíció információk alapján. Ahhoz, hogy ezt meglehessen valósítani, szükség van egy állandó TCP kapcsolat fenntartására, cserébe azonnal tudunk értesítést küldeni egy adott cellában elhelyezkedő összes felhasználónak. A PNS-ben minden cellához nyilvántartunk különböző üzeneteket (NotificationMessage) érvényességi időkkel. Ha egy résztvevő átlép az egyik cellából a másikba, akkor megkapja az új cellához hozzákapcsolodó üzeneteket, illetve ha éppen beállítanak egy üzenetet ahhoz a cellához ahol éppen tartózkodik, azt azonnal megkapja.

Ahogy az korábban már leírtuk, amikor egy mobil alkalmazás beküld egy mérési adatot a szervernek az először a



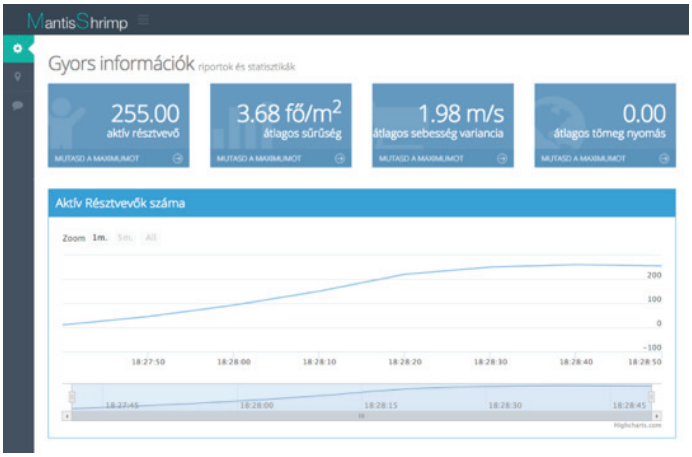
6. ábra: Térképes megjelenítés a felhasználói felületen. A szimuláción egy koncert megfigyelése látható



5. ábra: A TrafficAggregatorService áteresztőképességének. Másodpercenként hány üzenetet képes továbbítani a feldolgozó szerverek felé.

TrafficAggregatorService-be érkezik be. Hogy ne legyen felesleges kommunikáció eszköz és szerver között a beérkezett mérési adatokat a TAS átalakítja egy pozíció üzenetté és ezt továbbítja PNS-s felé, hiszen egy mérési adat tartalmaz minden szükséges információt, hogy eljuttassuk a felhasználónak a megfelelő üzeneteket, nem szükséges külön elküldeni a PNS felé.

Mivel a PNS-nek állandó kapcsolatot kell fenntartania minden aktív eszközzel, úgy kellett megtervezni, hogy rengeteg kommunikációs csatornát tudjon szimultán kezelni. A 2. ábrán látható, hogy két fő komponens található a PNS-en belül. A mobil eszközökkel a PushNotificationServer-ek (PS) tartják közvetlenül a kapcsolatot Netty segítségével, így egy ilyen szerver képes több tízezer felhasználó kezelésére is. Természetesen nagy számok esetén célszerű, hogyha több példány is fut belőlük így osztva el a terhelést. Hogy a PS-ek működése összehangolt legyen szükséges volt még egy központi vezérlő elemre is, a PushNotificationController-re (PNC). Ennek feladata a nevéből is adódóan, hogy egyrészt fogadja a TAS felől érkező pozíció adatokat és továbbítja a megfelelő PS felé, ami kapcsolatban áll az adott felhasználóval, valamint hogy kezelje a szervezőktől,



7. ábra: A Gyors információk felület.

rendfenntartóktól érkező üzenetek beállítását vagy esetleges törlését.

A PNS fontos funkciója, hogy rajta keresztül továbbítjuk az aktuális térképet is (cella felosztást), mint egy inicializáló üzenet lenne. Így megoldható, hogy a térképeket nem kell előre beletölteni az mobil alkalmazásokba, hanem azok dinamikusan képesek működni.

D.Felhasználói felület

A felhasználói felület kialakításánál egyrészt cél volt az, hogy a lehető legegyszerűbb formában, logikusan jelenítse meg a kinyert információkat, azt a célt szolgálva, hogy a szervezők, rendfenntartók minél gyorsabban átlássák a kialakult helyzetet, legyen egy jó rálátásuk az aktuális állapotra. Másrészt fontos volt az is, hogy a felületet könnyen el lehessen érni és sok eszközzel legyen kompatibilis. Ezért esett a választás egy webes felületre. Manapság rengeteg olyan ingyenes elérhető plugin található meg, melyek segítségével relatíve gyorsan fejleszthetünk jól átlátható felületeket.

A felhasználói felületet egy Apache HTTP Server szolgálja ki, a háttérben pedig egy php-ban írt webes alkalmazás kezeli le a kéréseket. Hogy a felhasználói felület alkalmazkodjon a mobil eszközök különféle felbontásához is, a Twitter Bootstrap-et használjuk.

Hogy a felület elemek mindig az aktuális állapotot tükrözzék, a böngésző periodikusan lekérdezéseket küld a webszerver irányába. Ezekben a lekérdezésekben csak státusz azonosítók közlekednek. A szerver összehasonlítja az adatbázisból elérhető aktuális állapotot a böngészőből érkezettel. Ha nincs egyezés a kettő között, szól a böngészőnek, hogy új adatokat lehet letölteni. Ezután a letöltött adatokat felületelemtől függően jeleníti meg.

Gyors információk

Ha belép egy szervező az oldalra ez fogadja kezdő képernyőként. A felületen gyors statisztikai adatokat kaphat a tömeg aktuális állapotáról, illetve egy idővonalon követheti, hogy hogyan változtak a különböző állapotokban a mértékek.

Térkép

Ezen az oldalon a szervező térképen láthatja az aktuális sűrűségi és tömegnyomási adatokat. A lejátszás gombra kattintva képes megtekinteni, hogy az időmúlásával hogyan változtak az állapotok, de bármikor megállíthatja, kereshet benne, ha úgy kívánja. Ha a térképen rákattint egy cellára egy felugró ablakban megjelenik a celláról elérhető összes historikus adat is.

PushNotificationService

Ezen a felületen a szervezőknek lehetősége nyílik arra, hogy adott cellákhoz különböző üzeneteket állíthassanak be. Az új üzenet hozzáadása gombra kattintva egy felugróablakban az üzenethez be kell állítani címet, érvényességi időt, cellát és tartalmat. Ezután a beállítás gombra kattintva az üzenet azonnal mentődik és azonnal meg is jelenik az összes olyan eszköznek, amely az adott cellában tartózkodik vagy fog tartózkodni. A le nem járt üzenetek az ábrán látható táblázatban jelennek meg. Ha esetleg a lejárati idő előtt szeretnének törölni egy üzenetet azt a Törlés gomb segítségével tudják megtenni.

IV. Szimuláció

A szimulációk során az volt a fő célunk, hogy megbizonyosodjunk arról, rendszerünk helyesen működik-e. Az egyes komponenseket külön-külön is komolyabb mérések alá vetettük, hogy megbizonyosodjunk azok megbízható, stabil működéséről.

A tesztek elvégzéséhez szükség volt egy eszköz szimulátor elkészítésére is, ami képes arra, hogy szimuláljon egy vagy több eszközt, amely tökéletesen képes kommunikálni a szerverekkel, tehát ahogy képes szimulált eredményeket beküldeni, ugyanúgy képes üzenetek fogadására is egy PushNotificationServer-től. A validációs teszteket sikeresen végrehajtottuk, illetve a TrafficAggregatorService teljesítményét is sikerült korlátozottan megmérnünk.

A TAS tesztjeit localhost-on végeztük egy 2013-as Macbook Air-en (CPU: 1.3GHz i5, Memória: 4GB 1600Mhz DDR3). Vizsgálataink során egyrészt arra voltunk kíváncsiak, hogy másodpercenként mennyi üzenetet képes torlódás nélkül továbbítani a TrafficAggregator az eszközök felől a kiértékelő szerverek felé. Azt tapasztaltuk, hogy másodpercenként körübelül 3500-4000 üzenetet is képes továbbítani, viszont megfigyeléseink alapján ez a teszthardver teljesítőképességének a felsőhátra, egy erősebb környezetben valószínűleg ennél is magasabb eredményeket érhetnénk el.

Lemérésre került, hogy mennyi idő alatt áramlik át a TAS komponensen keresztül egy üzenet, lényegében mekkora késleltést ad hozzá a TAS a mérési adat beérkézéséhez. Itt igen érdekes jelenséget figyeltünk meg: 1 millió szimulált üzenetet küldve körübelül az első 1-2 ezer üzenet esetén magas 60-200 ms késleltetési időket figyeltünk meg, majd ez hirtelen lecsökkent és utána ez az érték már nem is emelkedett meg szignifikánsan. Átlagosan 120-140 µs közötti késleltetést mértünk.

V. Konklúzió

Manapság a tömegrendezvények óriási látogatottságnak örvendenek, ugyanakkor számtalan veszélyt is hordoznak magukban. Célunk egy olyan tömegfelügyeleti rendszer elkészítése volt, melynek segítségével a biztonságért felelős szervek, közel valós időben, egy jól átlátható felületen keresztül átfogó képet kaphatnak a tömeg helyzetéről, mozgásáról.

A jövőben szeretnénk új és összetettebb szimulációkat végrehajtani, ezek eredményét felhasználva fejleszteni és optimalizálni a rendszer magfunkcióit. Továbbá tervezzük új funkciók elhelyezését a felhasználói felületen, annak érdekében, hogy szervezők még több perspektívából vizsgálhassák meg a tömeg állapotát. A későbbiekben tervezzük azt is, hogy a rendszert kiegészítjük egy aktívabb menedzsment rendszerrel, melynek segítségével az egyes komponensek működése monitorozható és dinamikusan konfigurálható lesz.

HIVATKOZÁSOK

[1] Security NewDesk: How many cameras in the UK? Only 1.85 million, claims ACPO lead on CCTV, online: http://www.securitynewsdesk.com/how-many-cctv-cameras-in-the-uk/, letöltve: 2015 Szeptember 7

[2] Donna Nelson, Traffax Inc., Riverdale MD: Proximity Information Resources for Special Event, (2012 Június)

[3] Traffax Inc.: BluFAX Concept, online: http://www.traffaxinc.com/content/blufax-concept, letöltve: 2013 November 21

[4] Jie Yin, Andrew Lampert, Mark Cameron, Bella Robinson, and Robert Power: *Using Social Media to Enchance emergency Situation Awareness*, IEEE Intelligent Systems, vol. 27, no. 6, 2012, pp.52-59

[5] Eve Mitleton-Kelly, "Co-evolution of Intelligent Socio-technical Systems", 2013, pp.61-77

[6] Helbing, D., Johansson, A., Al-Abideen, H.Z.: Dynamics of crowd disasters: an empirical study. Phys. Rev. E 75(4), 046109 (2007)

[7] Andreas Schadschneider, Wolfram Klingsch, Hubert Klüpfel, Tobias Kretz, Christian Rogsch , Armin Seyfried: Evacuation Dynamics: Empirical Results, Modeling and Applications, *Springer* (2008)

[8] Norman Maurer, Marvin Allen Wolfthal, "Netty in Action", 10. kiadás, Manning Publications, 2014

[9] Marko Bonaći, Petar Zečević, "Spark in Action", 5. kiadás, Manning Publications, 2015

Képosztályozás emberi és gépi tanulás esetén

Papp Dávid
Távközlési és Médiainformatikai Tanszék,
Budapesti Műszaki és Gazdaságtudományi Egyetem,
Magyar Tudósok krt. 2., H-1117, Budapest, Magyarország
pd948@hszk.bme.hu / pappdavid27@gmail.com

A gépi tanulás és az emberi tanulás összehasonlítása izgalmas téma, melynél párhuzamba állítható a két folyamat. A gépi tanuláshoz nagy mennyiségű címkézett tartalomra van szükség, melynek előállítása költséges, míg az emberi tanulásnál elég néhány, vagy akár egyetlen mintakép egy séma elsajátításához. A két folyamat közti alapvető és egyben legnagyobb különbséget a priori információk eltérő mértéke képezi. Korunk egyik alapvető célja, hogy a gép által elérhető eredményeket maximalizáljuk, így rengeteg emberi erőforrás takarítható meg. A tanulási folyamatok eredményességét képi tartalmak osztályozásával mértem, melyet iteratívan végeztem az emberi és gépi esetek összehangolásához.

Kulcsszavak—emberi tanulás; gépi tanulás; képosztályozás

I. Bevezetés

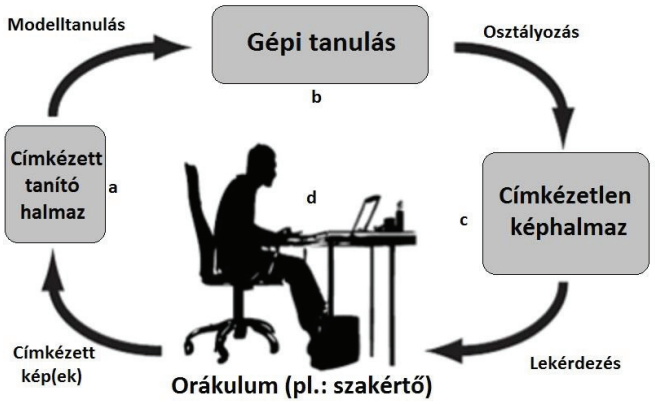
Manapság, a digitális világ növekedésének köszönhetően rengeteg fénykép található különböző adathordozókon, szerte a világban. Az interneten fellelhető képi tartalmak száma pedig megbecsülhetetlen, így korunk egyik alapvető problémája ennek a hatalmas adatmennyiségnek a rendszerezése. Ez rendkívül fontos számtalan alkalmazás számára, amelyek képekkel foglalkoznak, gondoljunk csak egy képkereső programra, vagy képnézegetőre. Ezen alkalmazások többféle módszert használnak a felhasználók segítése érdekében, hogy minél egyszerűbben találjanak számukra minél relevánsabb fényképeket, valamint több típusú részfeladatot kell megoldaniuk, mint például a képek osztályozása, rangsorolása, csoportosítása.

Vizuális információk alapján, emberi szemmel szinte tévesztés nélkül lehet képeket kategorizálni, viszont ez nagyon időigényes és (emberi) erőforrás igényes feladat. Ezzel szemben a számítógép kontextusában egy kép pixeleket, színeket, alakokat, textúrákat tartalmaz, melyekből meghatározni azt, ami egy ember számára egyértelmű egy képre ránézve, közel sem triviális feladat. Nem ritka, hogy több száz kategória közül kell eldöntenie egy képosztályozó algoritmusnak, hogy melyik az az egy vagy néhány, amelyikbe az adott kép beleillik. Alapvető probléma az is, hogy hasonló szemantikus információval bíró képek is lehetnek nagyon különbözőek.

Tehát a szemantikus képosztályozásnak két fontos és nehéz folyamata van, az első a képek olyan módú reprezentálása, amely lehetővé teszi azok összehasonlítását, a második pedig a képek osztályozása ezen kinyert információk alapján. A képosztályozási eljárások által elért eredmények általában gyengék, még bonyolult, nagy futási időt igénylő algoritmusok használata esetén is, mivel a fentebb említett okokból adódóan nagyon összetettek és rengeteg a hibalehetőség. Körülhatárolt területeken viszont jó eredménnyel alkalmazhatóak, mint például az orvosi diagnosztika, vagy a karakterfelismerés.

Az emberi és gépi tanulás összehasonlításához egy konkrét mérési tervet dolgoztam ki, valamint több különböző képhalmazt állítottam össze. Az emberi tanulás eredményének méréséhez egy böngészőben futó alkalmazást használtam, melyben a felhasználó személyek osztályozni tudták a képeket. Egy képhalmaz osztályozásának kezdetén, minden egyes kategóriából megadtam egy mintaképet, majd véletlenszerűen következtek a képhalmaz további képei, melyeket a megfelelő osztályban kellett elhelyeznie a felhasználónak. Minden egyes kép osztályozása után megadtam annak valós címkéjét, így kiderült, hogy helyes volt-e a döntés. A gépi tanulás mérését hasonlóan oldottam meg, így valóban összehasonlíthatók az eredmények. A folyamat az 1. ábrán tekinthető meg. Minden iterációban egy új, ismeretlen képet osztályozok, majd kiegészítem azzal a tanító halmazt, és újratanítom a rendszert a következő kép osztályozása előtt. Ezzel azt érem el, hogy a tanítórendszer is folyamatosan bővíti tudását, akár csak a felhasználó, miközben osztályozza a képeket. A lépéseket addig folytatom, míg minden képet fel nem címkéz az orákulum. Minél „később” osztályozok egy képet, annál több tanító képből tanul a rendszer. Ezért fontos kérdés, hogy a lépéseknél mikor melyik képet válasszam ki tesztelésre, majd címkézésre, és itt a hangsúly a címkézésen van. Megkülönböztetünk *passzív* és *aktív tanulási* módszereket, melyek a kiválasztás sorrendjének felállítási módjában különböznek.

Passzív tanulás esetén egy véletlenszerű, statikus sorrendet határozunk meg, míg aktív tanulás során a tanulórendszernek lehetősége van a tanuló adathalmaz egyes mintáit dinamikusan kiválasztani [1]. Az emberi és gépi tanulást passzív esetben hasonlítottam össze, hiszen az emberi aktív tanulás hasonlóan precíz mérése nehezen oldható meg. A képhalmazok mellé meghatároztam azt a statikus sorrendet, mely szerint a címkézetlen képek tesztelése történik, mind emberi, mind gépi esetben, így ezen a ponton is megegyezik a két folyamat. A következő fejezetben a képek matematikai reprezentálásának módját ismertetem, majd az osztályozás folyamatát tárgyalom. Végül az emberi és gépi tanulás összehasonlításának kísérletét és annak eredményeit mutatom be.



1. ábra: A gépi tanulás mérése: címkézett tanító halmaz (a), címkézetlen képhalmaz (c, teszthalmaz), gépi tanuló algoritmus (b), *orákulum* (d)

II. KÉPEK REPREZENTÁLÁSA

A. *Vizuális kódszavak*

Az eljárásom alapját egy általános, hasonló feladatokban gyakran használt módszer képezi, ami nem más, mint a szózsák modell (*Bag of Words*) képeken alkalmazott változata [2, 3]. Az elképzelés lényege, hogy egy képet a rajta szereplő vizuális elemek, más néven *vizuális kódszavak* összességével reprezentálunk, és ezek térbeli elhelyezkedésétől eltekintünk. Tehát egy képet csupán a vizuális kódszavak eloszlásával jellemzünk, és ebből próbálunk meg következtetni a szemantikus tartalmára. A vizuális kódszavak meghatározásához úgynevezett *alacsony szintű leíró*kra (röviden: leíró) van szükségünk, melyek a kép egy adott pontjának környezetében előforduló lokális tulajdonságokat foglalják magukban. Ezek lehetnek színek, textúrák, alakok és még sok más. Az algoritmus első feladata tehát, hogy meghatározza azokat a pontokat, melyek érdemesek leírók számítására, más néven a *kulcspontok*at.

A kulcspontok meghatározásához a Harris-Laplace detektort használtam, melynek alapja a Harris detektor, vagy más néven kombinált sarok és él detektálás [4, 5]. Az eljárás lényege, hogy egy képen azonosítja az éleket, ahol a kép pontjainak intenzitásértékeiben nagy változás következik be. Két él találkozásánál (sarokpontnál) pedig az intenzitásváltozás, azaz a derivált, két irányban is nagy abszolút értékű lesz. Ennek meghatározására egy csúszó ablakot vizsgál. A Harris detektort K. Mikolajczyk és C. Schmid fejlesztette tovább úgy, hogy az invariáns legyen a skálázásra is, ezt nevezik Harris-Laplace detektornak [6].

A második főbb lépés, hogy minden kulcspontot SIFT (Scale Invariant Feature Transform) leíróval jellemzek, melyet David G. Lowe fejlesztett ki [7]. Ez tulajdonképpen egy 128 dimenziós vektor, amely egy pont lokális környezetében lévő gradiensek orientációjából számolható. Minden kép minden kulcspontjában kiszámítom ezeket a leírókat.

Az egymáshoz hasonló leírók jelentenek egy vizuális kódszót. Ezek meghatározásához klaszterezem az összes képből kinyert SIFT leírót a GMM (Gaussian Mixture Model) módszer segítségével [8]. A GMM egy generatív módszer az alacsony leírók klaszterezésére, tehát az egyes klaszterekhez egy-egy valószínűségi modellt rendel. A pontok valószínűségi sűrűségfüggvényét K darab Gauss-féle függvény súlyozott összegével írja le:

$$p(X \mid \lambda) = \sum_{j=1}^{K} \omega_j g(X \mid \mu_j, \sigma_j) \qquad (1)$$

Itt az $X=\{x_1,x_2,...,x_N\}$ jelöli az M dimenziós adatvektorokat (SIFT leírók), λ a keresendő paramétert, $g(X|\mu_j\sigma_j)$ a Gauss-féle függvényeket (ahol μ_j a várható értéket, σ_j a szórást jelöli), valamint ω_j pedig a súlyozó együtthatókat. A GMM λ paraméterét ML (Maximum Likelihood) becsléssel határozzuk meg. Az x_i adatvektorok közt függetlenséget feltételezve a következőt írhatjuk fel:

$$p(X \mid \lambda) = \prod_{i=1}^{N} p(x_i \mid \lambda) \qquad (2)$$

Ez a kifejezés nem-lineáris a λ paraméterre nézve, tehát a direkt maximalizálása nem lehetséges. Erre egy speciális iteratív módszert szokás használni, az EM (Expectation Maximization) algoritmust [9, 10].

A GMM eredményeként megkapom a vizuális kódszavakat (az egyes Gauss függvények), melyekre tekinthetünk úgy, mint a képhalmaz tömör reprezentálására. A célom az, hogy minden képet jellemezni tudjak a tartalma alapján, méghozzá olyan módon, hogy azok összehasonlítása, megkülönböztetése, osztályozása lehetővé váljon. Így mindenképpen szükségem van egy olyan magasabb szintű képi leíró megalkotására, amely nem csak egy kulcspontot jellemez, hanem egy teljes képet. Erre a feladatra a Fisher-vektor használom.

B. *Fisher-vektor*

A Fisher-vektor a képfeldolgozási témakörök közül a képosztályozásban a legelterjedtebb így az én algoritmusomba tökéletesen beleillik [11, 12]. Ez egy rendkívül komplex leíró, amely arra használható, hogy egy teljes képet jellemezzünk vele egyetlen vektor formájában. Kiszámításához szükség van a SIFT leírókra, illetve a vizuális kódszavakra, tehát a GMM-re. Valójában azt fogja ez megmondani, hogy egy kép milyen eloszlásban tartalmaz vizuális elemeket, ahol ezek a vizuális elemek a képhalmaz egészéből kerülnek ki.

A GMM tárgyalásánál bevezetett jelöléseknek megfelelően legyen $p(X|\lambda)$ a valószínűségi sűrűségfüggvény ($\lambda=\{\omega_j,\mu_j,\sigma_j|j=1...K\}$), legyenek $X=\{x_1,x_2,...,x_N\}$ az M dimenziós adatvektorok, melyek itt nem az összes SIFT leírót jelentik, csak azokat, melyek egy adott képre vonatkoznak. A sűrűségfüggvény gradiense, azaz a logaritmusának deriváltja megadja, hogyan írja le legjobban a modell az adatvektorokat, tehát a képet, így tulajdonképpen ez a mennyiség a Fisher-vektor:

$$\nabla_\lambda \log p(X \mid \lambda) \qquad (3)$$

Ebből következik, hogy a Fisher-vektor összesen $K(2M+1)-1$ dimenziós (-1, mivel egy súlyozó együttható kiszámítható a többi ismeretében). Az én implementációmban $K=256$, mivel ennyi Gauss-függvényt definiálok $M=128$, mivel a SIFT leírók 128 dimenziósak. Ez azt jelenti, hogy az én esetemben egy Fisher-vektor 65791 dimenziós. A Fisher-vektorok tehát egységes és egyértelmű reprezentációi a képeknek és ezeket a vektorokat fogom felhasználni az osztályozáshoz.

III. OSZTÁLYOZÁSI FOLYAMAT

Ebben a fejezetben a képosztályozás folyamatát fogom bemutatni, mely során passzívan tanul az algoritmus. Jelölje T a teljes képhalmazt, amelyet osztályozni szeretnék, és jelölje S azt a kiindulási képhalmazt, mely címkéi a tanulórendszer számára már ismertek. A célunk, hogy T minden elemére adjunk egy jóslatot a kategóriájára vonatkozóan. Ehhez dinamikusan fogom kezelni mind a T, mind pedig az S halmazt. Kezdetben a tanulóhalmazunk S, és a teszthalmazunk T. Lépésenként T-ből kiválasztunk és osztályozunk, majd ki is törlünk egy képet és ezzel a képpel bővítjük az S halmazt, így a rendszer tudása egyre növekszik, és várhatóan pontosabb becslést tud majd adni a következő kép(ek)re. Fontos megjegyezni, hogy ilyenkor a képpel együtt annak valódi osztálycímkéjét is átadom a tanulórendszernek. Passzív tanulás esetén, az S bővítése nem szabályozott, hiszen egy véletlenszerűen generált, statikus sorrend alapján kerülnek át az egykori tesztképek a tanulóhalmazba. Ahogy már korábban említettem, ez a sorrend egységes az emberi és gépi tanulás esetén. Nézzük, hogyan történik egy kép osztályozása.

A képek reprezentálásával megkaptam a képek magas szintű leíróit (Fisher-vektorok) $\varphi_1,\varphi_2,\ldots,\varphi_N$, valamint azok valós osztálycímkéi, melyeket az angol elnevezésnek megfelelően *ground truth*-nak nevezünk. Egy teszt képet a magas szintű leírója alapján osztályozok úgy, hogy minden c kategóriához meghatározok egy $f(\varphi)$ értéket, amely azt fogja megadni, hogy az adott kategória mekkora bizonyossággal jellemző a képre. Ezt úgy teszem meg, hogy létrehozok c darab egymástól független bináris osztályozót, melyekkel külön-külön elvégzem a tanítást és az osztályozást. Minden tanításnál a kiválasztok egy osztályt, és abba tartozó képek lesznek a pozitív minták, az összes többi pedig negatív minta. Ezt *one-against-all* módszernek nevezzük, és egyszerűsége ellenére rendkívül jól használható. Bináris osztályozóként az SVM (*Support Vector Machine*) egyik változatát használom amelyet C-SVC osztályozónak neveznek [13, 14]. A módszer lényege, hogy a végtelen sok szeparáló hipersík közül a maximális margójút találja meg.

IV. TESZTELÉS ÉS EREDMÉNYEK

A. *Felhasznált képhalmazok*

A teszteléshez öt különböző képhalmazt használtam, melyek mindegyike manuálisan kigyűjtött és címkézett képekből áll. Mivel célom az emberi és gépi tanulás összehasonlítása, ezért viszonylag kisméretűek a teszthalmazok, hogy az emberi tanulás kiértékelésében részt vevő személyek számára ne legyen túl megterhelő a rengeteg kép osztályozása. Az 1. táblázatban összefoglaltam ezek méretét, a definiált kategóriák számát, illetve a tesztelő személyek számát.

1. táblázat: Teszteléshez használt képhalmazok adatai

	Méret	Kategóriák száma	Tesztelő személyek száma
Képhalmaz 1	100	7	12
Képhalmaz 2	100	8	9
Képhalmaz 3	116	6	12
Képhalmaz 4	105	5	6
Képhalmaz 5	106	6	7

A kiindulási címkézett halmazhoz minden osztályból választottam egy-egy képet, ez alapján kezdték mind a tesztelő személyek és az algoritmus a tanulást. Az osztályozáshoz megadtam a megfelelő méretű szekvenciákat, melyek a címkézetlen képek érkezési sorrendjét határozzák meg, ez szintén egységes volt a webes felület és a képosztályozó rendszer esetén.

B. *Kiértékelés menete*

Az osztályozások eredményeit a pontosság (*accuracy*) szempontjából értékeltem ki, melyhez szükség volt a tesztelt képek során született döntések eredményeire, hogy sikeres volt-e a kategóriába sorolás, vagy sem. A tesztképekhez tartozó szekvencia mellé egy 0/1 döntési eredményt rendeltem, így minden képről eltároltam, hogy annak osztályozását az adott tesztelő entitás (mely lehet gépi vagy emberi) miként végezte. Ez által, a pontosság meghatározása a következő képlet alapján történik:

$$accuracy = \frac{döntési\ vektor\ 1-eseinek\ száma}{szekvencia\ mérete} \tag{4}$$

A folyamatosan bővülő tanulóhalmaz egyre több információt ad a tesztelőnek, így elméletben, a fennmaradó címkézetlen képeket nagyobb pontossággal képes osztályozni, mint a korábbiakat. Ennek mérése érdekében egy további mutatót is kiszámítok, melyhez egy csúszó ablakot mozgatok a szekvencia elejétől a végéig, és mindig az *ablak alatti* szekvencia által kijelölt képek osztályozási *pontosság*át határozom meg:

$$accuracy_w = \frac{ablak\ alatti\ 1-esek\ száma}{ablak\ mérete} \tag{5}$$

Ahogy az 1. táblázatban látszik, minden képhalmazt több személy tesztelt, így a kapott pontossági értékeket átlagoltam. A következő alfejezetben a kapott eredményeket mutatom be, és hasonlítom össze. A kapott átlagos ablak alatti pontosságokat vonal diagramok segítségével ábrázolom, a tanított képek számának függvényében (tehát a szekvencián előre haladva). Az ablak méretét 10-re választottam, így a kiértékelés elején, a 10. beérkezett kép után kapom meg az első pontossági értéket, és az ábrázolás onnantól kezdődik az utolsó beérkező képig

C. *Eredmények I*

Az első képhalmaz különböző film poszterekből állt, a definiált kategóriák pedig az egyes filmek műfajai voltak (pl.: animációs, akció, vígjáték, horror, stb.). Ez azt jelenti, hogy az egyes osztályokba tartozó képek nem kizárólag vizuális hasonlóság alapján kapcsolódnak (lásd 2. ábra). Éppen ezért a képosztályozó algoritmusnak nehéz dolga volt ezzel a teszthalmazzal, hiszen az egyedül a tartalmi egyezéseket veszi figyelembe, míg az emberi agy könnyedén megtalálja a mögöttes jelentésből fakadó kapcsolatot is.

Ahogy a 2. ábrán látszik, az egyes kategóriánként megjelenített képek közt nem lehet egyértelmű vizuális kapcsolatot meghatározni. A gépi tanulás eredménye a 4. ábra (a) diagramján látható, az emberi tanulással kapott eredmények pedig a (b) diagramon tekinthetők meg.

2. ábra: Példa képek az első képhalmazból, (a)-(g)-ig oszloponként azonos osztályú képekkel

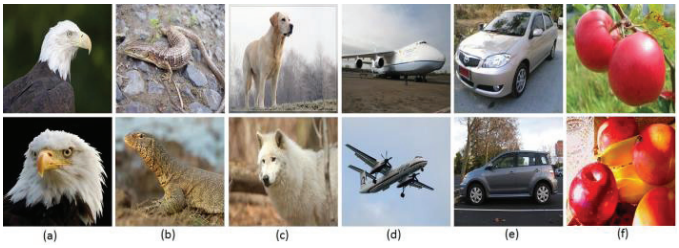
Jól látszik, hogy a gépi tanulás, sőt, az emberi tanulás esetén is a tanított képek számának növekedésével a pontosság is növekszik (kevés kivétellel). Az algoritmus a szekvencia elején érkező képeket nagy hibával osztályozta, viszont 70 kép után javul valamennyit a pontosság, viszont utána vissza is esik. Ezért azt állapítottam meg, hogy ezeket a kategóriákat nem sikerült eredményesen megtanulnia a gépnek. Az emberi tanulás láthatóan az első képtől kezdve jobban teljesít, viszont ez várható is volt, hiszen a gépnek nincs priori ismerete, emberek számára viszont sok segítséget nyújthatnak tapasztalati ismeretek, például ebben az esetben, ha a tesztelő látta már azt az adott filmet, melyet a poszter ábrázol. Ezért előfordult a 0,9-es ablak alatti pontosság is, míg gépi tanulás esetén ez a legjobb esetben 0,7 volt.

D. Eredmények II

A harmadik képhalmaz a leghasonlóbb a képosztályozási feladatoknál megszokott teszthalmazhoz. Az itt definiált osztályokban (sas, kutyafélék, hüllők, gyümölcsök, szárazföldi négykerekű járművek, repülőgépek) jelentős vizuális hasonlóság van jelen (lásd 3. ábra), így gépi tanulással hatékonyan kategorizálhatóak a képek. A 3. ábrán minden osztályból 2-2 képet jelenítettem, hogy szemléltessem a különbséget az első és harmadik teszthalmaz között. Jól látható, hogy a 2. ábra kategóriáival ellentétben most tényleg tartalmi hasonlóság adja a kategorizálás alapját, és tulajdonképpen az osztályozó rendszert ilyen képhalmaz tesztelésére készítettem fel, ami az eredményeken is látszik.

A kategóriák meghatározásnál annyi nehezítést alkalmaztam, hogy néhány esetben több különböző objektum is beletartozik ugyanabba az osztályba, tehát úgynevezett *superclass*-okat definiáltam. Például a gyümölcsök kategóriát a banán, narancs, alma, körte, barack, málna és áfonya alkotja, illetve a kutyafélék osztályt a farkas, róka és kutya objektumok alkotják, valamint ugyanez igaz a szárazföldi négykerekű járművek esetén is. Látni fogjuk, hogy ilyen esetben a felhasználók szinte tévesztés nélkül képesek megmondani az egyes képek valódi osztályát. Ez nem meglepő, hiszen például egy rókát egy narancstól előzetes tanítási folyamat nélkül is 100%-os biztonsággal és pontossággal vagyunk képesek megkülönböztetni. A 4. ábrán láthatóak a gépi (c), valamint az emberi (d) tanulás eredményei.

A gépi tanulás eredményeiből is jól látható, hogy ez egy olyan teszthalmaz, melyre az algoritmus fel van készítve, így a korábban jósolt javulás a pontosságban egyértelműen észrevehető, a tanított képek számának növekedésével.



3. ábra: Példa képek a harmadik képhalmazból, (a)-(f)-ig oszloponként azonos osztályú képekkel

Az első néhány képet hibásan osztályozza a rendszer, viszont nagyjából 30 tanító kép után stabilan 0,7-0,9 közé áll be az ablak alatti átlagos pontosság. A 4. ábra (c) diagramján látható hogy szinte végig 100%-os a pontosság, az elején volt egy, illetve később néhány tévesztés.

E. Eredmények összefoglalása

A tesztek jól jellemzik az emberi és gépi tanulás közti különbségeket. A legfontosabb különbség, hogy az emberek által birtokolt előzetes információ egy jó alapot nyújt az osztályok közti hasonlóság gyors, akár azonnali felismeréséhez. Igazán jól összemérni a kettőt akkor lehetne, ha olyan képeket tudnánk előállítani, melyek a tesztelő személyek semmilyen eddig tapasztalatához, ismereteihez nem köthetőek, így számukra is elengedhetetlen lenne az előzetes tanulás, ahogy a gépi tanulás esetén láttuk. Erre használhatnánk például absztrakt, gép által véletlen generált képeket.

A csúszó ablak segítségével jól mérhetővé vált az, hogy a pontosság hogyan változik az osztályozás során (főleg gépi esetben), viszont a teljes teszthalmazon számított pontossági érték (lásd 4. egyenlet) meghatározása nem képes a javulás megmutatására. Ennek oka, hogy a teljes döntési vektort használjuk a kiszámításhoz, figyelmen kívül hagyva azt, hogy a döntési vektorban (elemszámában) előre haladva az osztályozáshoz felhasznált tanulóanyag mérete növekszik. Ennek ellenére ezeket is kiértékeltem, és a 2. táblázatban összefoglaltam, ahol W, G és E rendre az ablak alatti, gépi és emberi pontosságtípusokat jelölik. Például $\max(\text{accuracy}_{W,G})$ az 5. egyenlet alapján számolt ablak alatti pontosságok közül a maximálist jelenti, az adott teszthalmazra, gépi tanulás esetén, míg átlag()-al a 4. egyenletben felírt képlet alapján kapott pontosságot jelölöm.

V. KONKLÚZIÓ

A kísérlet fő célja az emberi és gépi tanulás összehasonlítása volt, melyhez egy mérési tervet dolgoztam ki. Az emberi tanulást egy böngészőben futó alkalmazás segítségével mértem, míg a gépi tanuláshoz elkészítettem egy iteratív tanulóalgoritmus,

2. táblázat: Teszthalmazonkénti pontosság értékek összesítése

	Teszt 1	**Teszt 2**	**Teszt 3**	**Teszt 4**	**Teszt 5**
$\max(\text{accuracy}_{W,G})$	0,7	0,6	0,9	0,7	0,9
$\text{átlag}(\text{accuracy}_{G})$	0,269	0,217	0,664	0,290	0,420
$\max(\text{accuracy}_{W,E})$	0,9	0,989	1	0,983	1
$\text{átlag}(\text{accuracy}_{E})$	0,716	0,859	0,998	0,901	0,991

melyre azért volt szükség, hogy a lehető legpontosabban mintázzam a tesztelő személyek által végzett osztályozási folyamatot. Az eredményeket kiértékeltem, és egyező beállítások mellett gépi tanulást használva is leosztályoztam összesen 5 különböző teszthalmazt, majd ezek eredményeinek kiértékelése következett. Az eredményeket összehasonlítottam osztályozási pontosság szempontjából, illetve egy csúszó ablak segítségével a tényleges tanulási folyamat „gyorsaságát" is lemértem (főként a gépi tanulásnál látszott ez).

Az emberek előnnyel indulnak a géppel szemben, hiszen rengeteg előzetes információ áll rendelkezésünkre, melyektől képtelenség elvonatkoztatni, és úgy tekinteni egy új osztályozási feladatra, mintha a tanító képeken kívül semmilyen tudással nem rendelkeznénk. Ezért a gépi tanulást fejleszteni kell, hogy minél jobban meg tudja közelíteni az emberi képességeket. A jövőben összetett aktív tanulási stratégiákat fogok kidolgozni és megvalósítani, hogy minél kevesebb tanító mintával minél pontosabb osztályozást tudjak elérni. Az elkészült aktív tanuló rendszerrel az eddig bemutatott képhalmazokat újrateszteltem és az új tapasztalatokkal, eredményekkel egészítem ki a kiértékelést. A passzív és aktív tanulás szélesebb körű összeméréséhez további teszthalmazokat fogok gyűjteni, melyek nagyobb méretűek (itt már nem lesz emberi tanulás).

Referencia

[1] Burr Settles, Active Learning Literature Survey, Computer Siences Technical Report 1648, University of Wisconsin – Madison, 2009.

[2] L. Fei-Fei, R. Fergus, and A. Torralba: *Recognizing and Learning Object Categories*, Part1 – Single object classes: *Bag of Words models*, *Part-based models*, *and Discriminative models*, 2009, pp. 2-16.

[3] S. Lazebnik, A. Torralba, L. Fei-Fei, D. Lowe, C. Szurka: *Bag-of-Words models*, Lecture 9, 2012, pp. 1-32.

[4] C. Harris, M. Stephens: *A combined corner and edge detector*. In C. J. Taylor, editors, Proceedings of the Alvey Vision Conference, pages 23.1-23.6. Alvey Vision Club, September 1988. doi:10.5244/C.2.23.

[5] Kató Zoltán: *Sarokpontok detektálása*, Képfeldolgozás és Számítógépes Grafika tanszék SZTE. http://www.inf.u-szeged.hu/~kato/teaching/DigitalisKepfeldolgozasTG/07-CornerDetection.pdf (letöltés dátuma: 2014.10.07.)

[6] Mikolajczyk, K., Schmid, C.: *Scale & affine invariant interest point detectors*, International Journal on Computer Vision 60(1), 2004, pp. 63-86.

[7] Lowe, D. G.: *„Distinctive Image Features from Scale-Invariant Keypoints"*, International Journal of Computer Vision, 60, 2, pp. 91-110, 2004.

[8] Reynolds, D. A.: *Gaussian Mixture Models*, Encyclopedia of Biometric Recognition, Springer, Journal Article, February 2008. http://www.ll.mit.edu/mission/communications/ist/publications/0802_Reynolds_Biometrics-GMM.pdf (letöltés dátuma: 2014.10.07.)

[9] Carlo Tomasi: *Estimating Gaussian Mixture Densities with EM – A Tutorial*, Duke University. http://www.cs.duke.edu/courses/spring04/cps196.1/handouts/EM/tomasiEM.pdf (letöltés dátuma: 2014.10.07.)

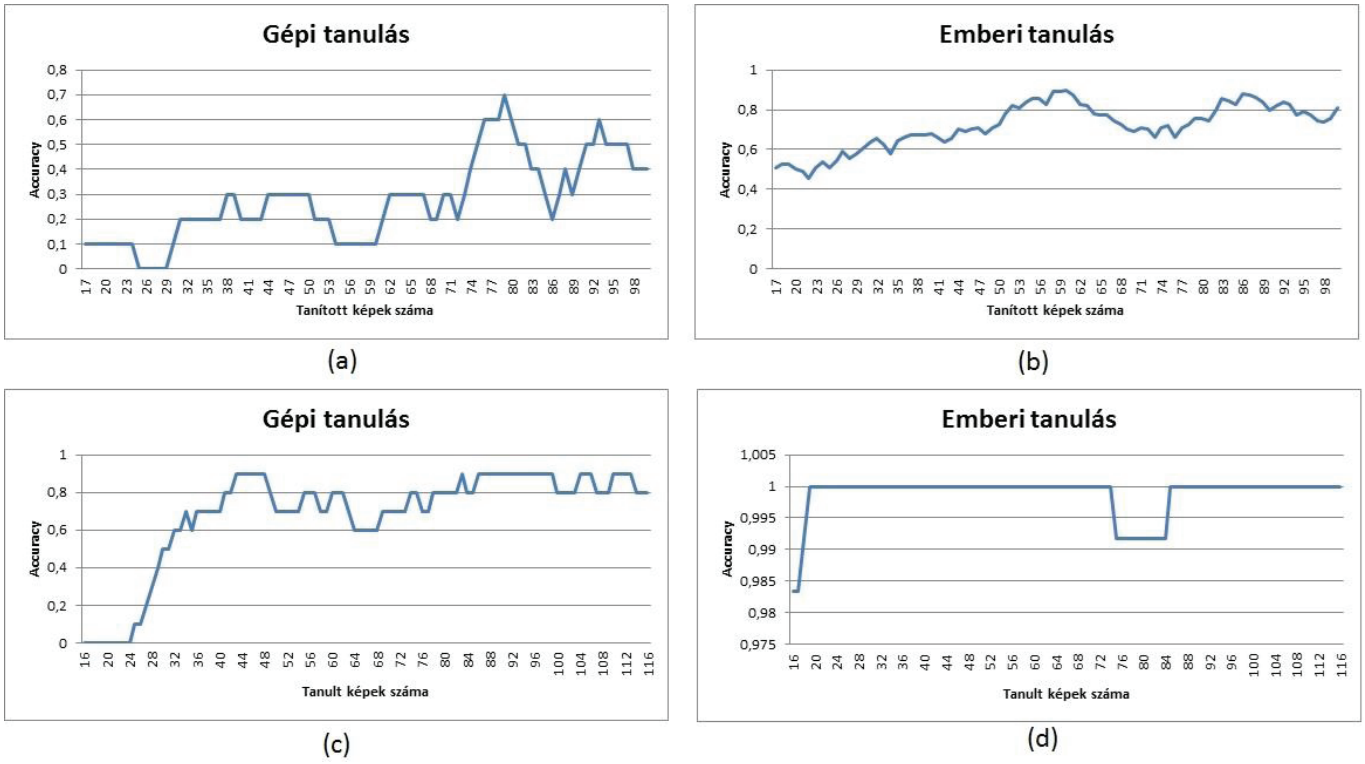
[10] Dempster, A., Laird, N., Rubin, D.: *Maximum Likelihood from Incomplete Data via the EM Algorithm*, Journal of the Royal Statistical Society 39(1), 1977, pp. 1–38.

[11] Jaakkola TS, Haussler D.: *Exploiting generative models in discriminative classifiers*. Advances in Neural Information Processing Systems (NIPS), Vol. 11, 1998, pp. 487-493.

[12] Florent Perronnin, Chris Dance: *Fisher kernel on visual vocabularies for image categorization*, CVPR, Computer Vision and Pattern Recognition, 2007.

[13] Boser, B. - Guyon, I. - Vapnik, V.: *A Training Algorithm for Optimal Margin Classifier*, Proc. of the 5th Annual ACM Workshop on Computational Learning Theory, 1992, pp. 144-152.

[14] Corinna Cortes, Vladimir Vapnik: *Support-vector networks*, Machine Learning, Volume 20, Number 3, 1995, pp. 273-297.



4. ábra: Az első képhalmaz tesztelése során kapott átlagos ablak alatti pontosságok a tanított képek számának függvényében; (a): gépi tanulás esetén, (b): emberi tanulás esetén. A harmadik képhalmaz tesztelése során kapott átlagos ablak alatti pontosságok a tanított képek számának függvényében; (c): gépi tanulás esetén, (d): emberi tanulás esetén

HÍRKÖZLÉSI
ÉS INFORMATIKAI
TUDOMÁNYOS
EGYESÜLET (HTE)

1051 Budapest, Bajcsy-Zsilinszky út 12.
Tel.: 353 1027; Fax: 353 0451
E-mail: info@hte.hu
www.hte.hu