پروژه کارشناسی:

کنترل ربات از راه دور با استفاده از اپلیکیشن اندروید

استاد راهنما: دکتر ولی درهمی

ارائه دهنده: فرزانه گرامی

گروه مهندسی کامپیوتر – دانشگاه یزد

تابستان ۹۷

فهرست

مقدمه:

امروزه ما می‌توانیم ببینیم که کارهایی که قبلا به صورت دستی کنترل می‌شدند اکنون با استفاده از ابزار، ماشین آلات و کنترل از راه دور به شکل خودکار انجام می‌شوند. همچنین امروزه رباتیک، جز یکی از ۱۰ صنعت برتر جهان محسوب می‌شود.کنترل ربات ها از راه دور و به دنبال آن استفاده از ربات در محیط‌های خطرناک، ویژگی دیگری است که باعث جذابیت ربات‌ها می‌گردد.

از طرفی دیگر در حال حاظر اندروید یکی از محبوب‌ترین و پر استفاده‌ترین سیستم عامل مورد استفاده‌ی دستگاه‌های هوشمند است و فناوری سخت افزاری مورد استفاده در دستگاه‌های هوشمند روز به روز بهبود می‌یابد. از این رو می‌توان گفت که استفاده از این پلتفرم برای کنترل سیستم‌های رباتیک مزیت بزرگی برای موارد استفاده عمومی یا صنعتی آن خواهد بود.

هدف اصلی این پروژه ساخت یک برنامه‌ی اندروید برای کنترل حرکت یک ربات با استفاده از اتصالات بیسیم است. بدین صورت که با توجه به تصویری که دوربین ربات می‌فرستد کاربر بتواند بر اساس واسط کاربری آماده شده ربات را از راه دور به مکان مورد نظر هدایت نماید.

معرفی بستر سخت افزاری و پروتکل‌های مورد نیاز:

بستر سخت افزاری:

به منظور مانیتور حرکات ربات در هنگام ارسال دستورات نیاز به انتخاب و راه اندازی دوربینی مناسب داریم. یکی از انتخاب‌های مناسب می‌تواند دوربین‌های مدار بسته تحت شبکه یا دوربین مدار بسته IP باشد.

دوربین تحت شبکه نوعی از دوربین است که معمولاً برای سامانه‌های حفاظتی و نظارت تصویری استفاده می‌شود و برخلاف دوربین‌های مدار بسته آنالوگ از پروتکل‌های شبکه برای ارسال اطلاعات استفاده می‌کند. این دوربین‌ها برای اولین بار توسط شرکت AXIS در سال ۱۹۹۶ معرفی شدند. دوربین مدار بسته شبکه در واقع نوعی کامپیوتر کوچک نیز محسوب می شود. این دوربین ها پس از دریافت اطلاعات تصاویر از حسگر آنها را به

اطلاعات دیجیتال تبدیل کرده و مطابق پروتکل های شبکه به شبکه می فرستند. دوربین‌های شبکه انعطاف‌پذیری بالایی دارند و آنها را می‌توان به راحتی در هر کجا از شبکه جابه‌جا کرد. از طریق دوربین‌های شبکه می‌توان از امکانات پردازش تصویر در صورت نیاز بر روی تصاویر استفاده کرد. پردازش‌هایی که روی تصویر انجام می‌شود به مدل دوربین و شرکت سازنده ی آن بستگی دارد. از جمله پردازش‌های تصویر : تشخیص عبور از خط، تشخیص ورود و خروج از منطقه حفاظت شده، تشخیص صدای محیط، تشخیص دود و حرارت و غیره را می‌توان نام برد.

در این پروژه ما از یک دوربین مدار بسته IP بولت و یک مودم همراه برای اتصال آن به شبکه استفاده کرده‌ایم. در اینجا باید به این نکته توجه شود که برای اتصال و دسترسی به دوربین از شبکه‌های خارجی نیاز به تهیه و تنظیم IP استاتیک وجود دارد (استفاده از IP استاتیک یکی از مطمئن‌ترین و راحت‌ترین روش‌های دسترسی به دوربین از شبکه‌های خارجی است) که باید در هنگام تهیه‌ی مودم در نظر گرفته شود.

شکل ۱: بستر سخت افزاری استفاده شده

پروتکل ONVIF:

دوربین های مدار بسته آنالوگ از نظر سازگاری با یکدیگر هیچ مشکلی نداشتند. اساسا دوربین‌های مداربسته و DVRهای مرسوم سنتی آنالوگ اگر از برندهای مختلفی هم باشند، باز هم بدون مشکل با هم ارتباط برقرار کرده و درکنار یکدیگر کار می‌کردند. در اوایل ظهور دوربین های تحت شبکه به دلیل کمبود تجهیزات و ضعف در سازگاری با یکدیگر، راه پیشرفت و گسترش‌شان ناهموار بود. شرکت‌های تولیدکننده تجهیزات نظارت تصویری تحت شبکه، پروتکل و قراردادهای خود را داشتند. از این رو لازم بود در یک پروژه همه تجهیزات را (شامل دوربین مدار بسته، دستگاه DVR و غیره) از محصولات یک برند انتخاب کرد. در ابتدا هر شرکت استاندارد و پروتکل‌های خود را برای انتقال و ضبط تصاویر داشت و این امر باعث عدم توسعه دنیای تازه متولد تجهیزات نظارت تصویری IP یا تحت شبکه می‌شد.

در سال ۲۰۰۸، شرکت‌های سونی، بوش و اکسیس پروتکل استانداردی به نام ONVIF (مخفف Open Network Video Interface Forum) را ایجاد کردند. پروتکل استاندارد ONVIF با هدف حل مشکل همکاری و ارتباط بین تجهیزات نظارت تصویری IP، به طوری که محصولات از شرکت‌ها و برندهای مختلف در کنار یکدیگر بدون مشکل کار کنند، شکل گرفت. استانداردی مشترک برای تسریع بخشیدن به توسعه و فراگیر شدن تجهیزات نظارت‌تصویری تحت شبکه.

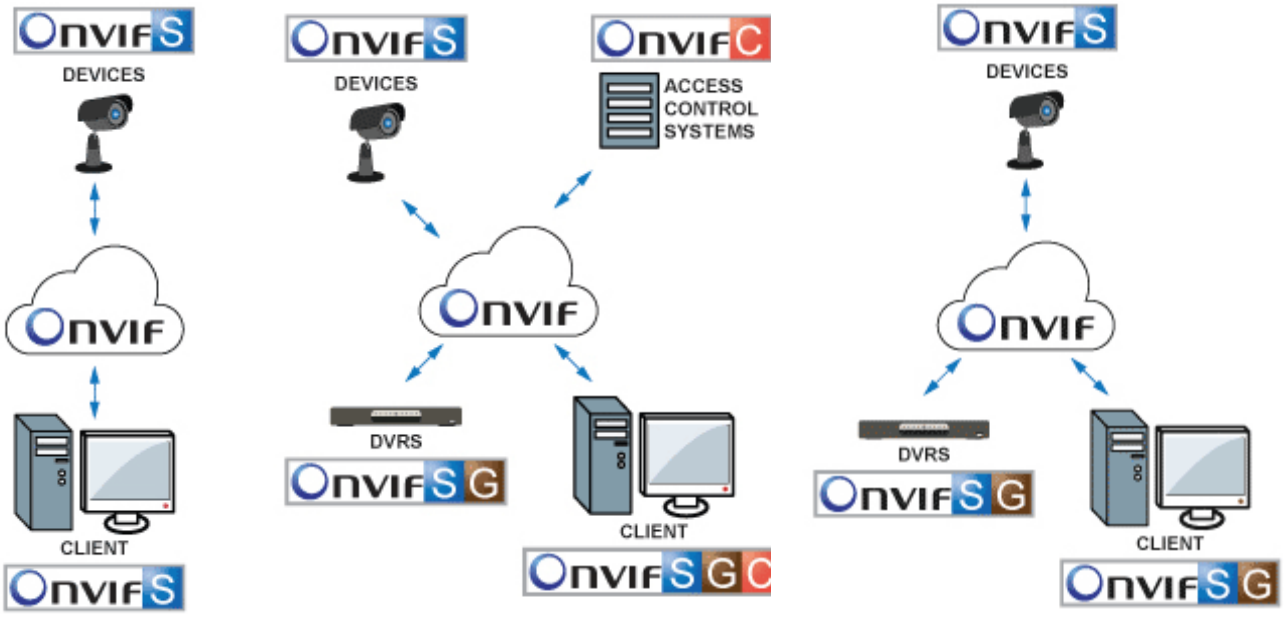
ONVIF علاوه بر استانداردسازی ارتباط بین محصولات تحت شبکه، همچنین باعث بهینه‌سازی روند فشرده‌سازی و انتقال اطلاعات صوتی و تصویری، یافتن دستگاه‌های IP روی شبکه، کنترل دوربین‌های PTZ، ورودی‌ها و خروجی‌های آلارم، روال‌های تنظیم و کنترل، موشن‌دیتکشن و غیره نیز شد. در کل پروتکل ارتباطی برای محصولات تحت شبکه مانند زبان برای انسان‌هاست. پروتکل ONVIF زبانی مشترک میان محصولات تحت شبکه است.

پروفایل‌های ONVIF:

هر پروفایل ONVIF دارای مشخصات فنی خاصی است که ایجاد ارتباط بین دستگاه‌ها با قابلیت‌ها و کارکردهای خاص را تضمین می‌کند. سازمان ONVIF در سال ۲۰۱۲ مفهوم "پروفایل" را معرفی کرد. وقتی

دستگاه‌ها و برنامه‌ها در یک پروفایل باشند، بی‌شک باهم سازگارند. پروفایل ONVIF به کاربر اجازه می‌دهد به راحتی ویژگی‌های یک پروفایل را بدون آگاهی از انطباق‌پذیری بین ورژن‌های ONVIF، تشخیص دهد.

- پروفایل S: جهت ارسال و انتقال تصویر (منتشر شده در سال ۲۰۱۱)
- پروفایل C: جهت کنترل دسترسی تحت شبکه ابتدایی (منتشر شده در سال ۲۰۱۳)
- پروفایل G: جهت ذخیره سازی و بازیابی (منتشر شده در سال ۲۰۱۴)
- پروفایل Q: جهت نصب سریع (منتشر شده در سال ۲۰۱۶)
- پروفایل A: جهت پیکربندی و تنظیم سیستم‌های کنترل دسترسی پیشرفته‌تر (منتشر شده در سال ۲۰۱۷)
- پروفایل T: جهت ارسال و انتقال تصویر پیشرفته (منتشر شده در سال ۲۰۱۸)



شکل ۲: پروفایل‌های ONVIF

پروتکل RTSP:

RTSP مخفف کلمه Real Time Streaming Protocol می‌باشد. به معنی پروتکل جریان بلادرنگ که برای انتقال سریع صدا و تصویر زنده یا ضبط شده مورد استفاده قرار می گیرد. این پروتکل در دوربین‌های IP کاربرد مهمی دارد. وقتی یک دوربین IP را به NVR متصل می‌کنید با توجه به آدرسی که قبلا توسط کمپانی

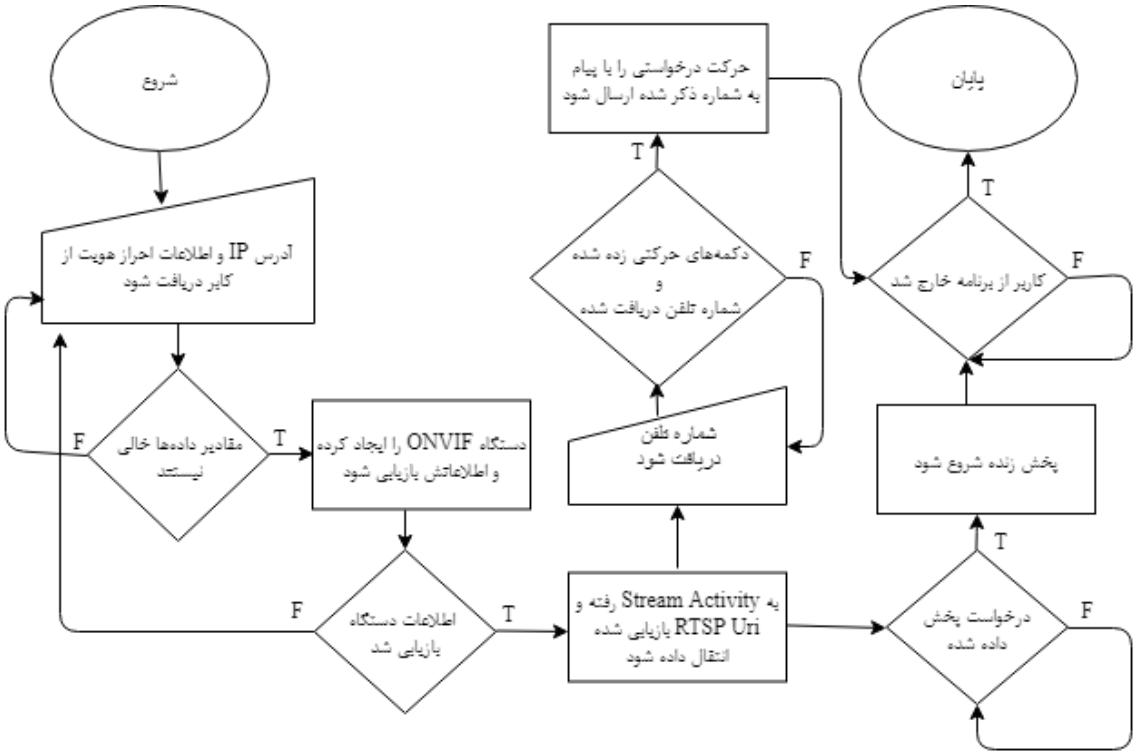
سازنده تعیین شده تصاویر زنده دوربین توسط این پروتکل از دوربین دریافت و خروجی NVR به شما نمایش می‌دهد.

این آدرس معمولا استاندارد خاصی ندارد و هر کمپانی معمولا به صورت اختیاری این آدرس برای دوربین تعریف می‌کند. در اکثر نرم افزارهای مدیریتی دوربین‌های IP و NVR های کتابخانه‌ای از این آدرس‌ها وجود دارد ولی با این حال در بعضی موارد نیاز می‌باشد این آدرس را شما به صورت دستی وارد کنید.

شرح ساختار و نحوه‌ی عملکرد برنامه:

شرح کلی:

روند کلی عملکرد این برنامه بدین صورت است که ابتدا اطلاعات مورد نیاز شامل IP، نام کاربری و گذرواژه‌ی دوربین از کاربر دریافت شده و با استفاده از آنها از طریق پروتکل ONVIF اطلاعات دوربین و آدرس RTSP مختص آن برای پخش تصاویر استخراج می‌شود سپس در حین پخش تصاویر زنده‌ی دوربین امکان ارسال دستورات حرکتی از طریق پیام (یا از طریق وب سرویس مورد نظر در صورت راه اندازی) وجود دارد.



شکل ۳: نمودار جریان برنامه

این روند در برنامه در دو activity انجام می‌شود:

- Activity اول (Main Activity) که دریافت اطلاعات کاربر، ارتباط با دوربین و استخراج آدرس در آن صورت می‌گیرد.
- Activity دوم (Stream Activity) که با گرفتن آدرس از activity اول امکان پخش تصاویر دوربین و ارسال دستورات حرکتی را دارد.

Main Activity:

برای استفاده از پروتکل ONVIF می‌توانیم از کتابخانه‌هایی که برای اندروید در این زمینه تهیه شده‌اند استفاده کنیم که برای این منظور خط‌های زیر را به build.gradle (Module: app) اضافه می‌کنیم تا کتابخانه‌ها اضافه شوند.

```
implementation 'com.rvirin.onvif:onvifcamera:1.1.6'
implementation 'com.squareup.okhttp3:okhttp:3.10.0'
```

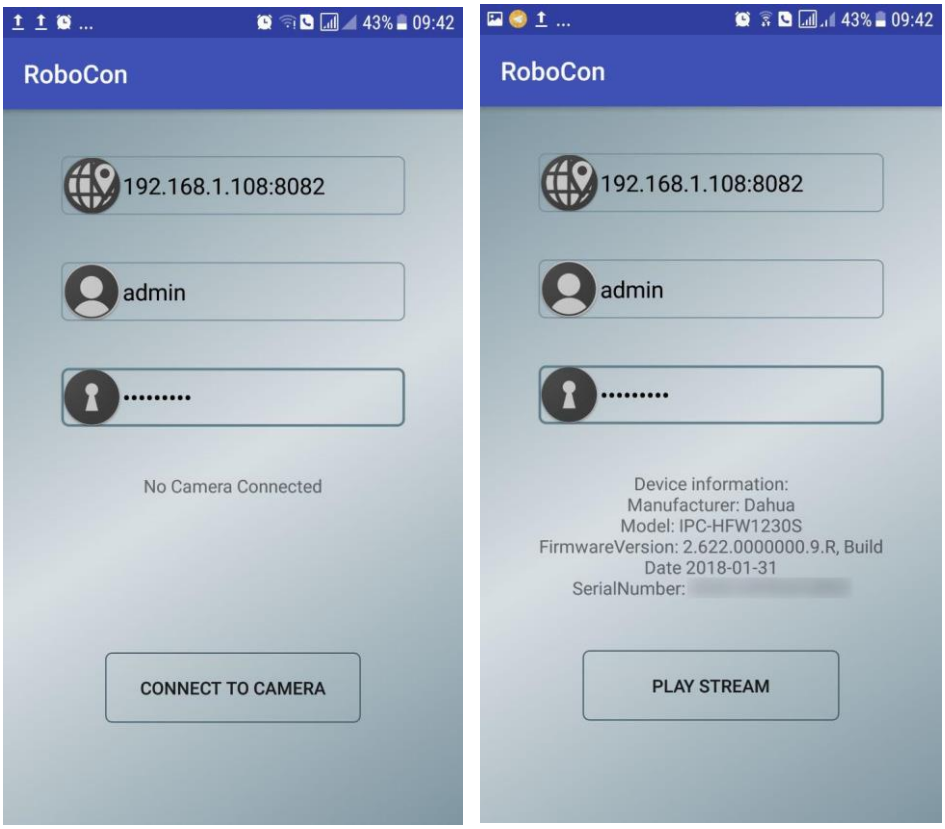
در ابتدا کاربر باید IP، نام کاربری و گذرواژه‌ی دوربینی که در نظر دارد به آن متصل شود را وارد کند (در صورتی که پورت http دوربین را تغییر داده شده است IP باید به همراه شماره پورت جدید وارد شود.) سپس ما با استفاده از آنها برای تعریف یک دوربین ONVIF در برنامه از کلاس onvifDevice استفاده می‌کنیم. این کلاس یک onvifListener دارد که تابع ()requestPerformed را تعریف می‌کند. این تابع وقتی صدا زده می‌شود که دوربین پاسخ درخواست‌های ما را بدهد (درخواست‌هایی مانند ,getServices getDeviceInformation, getProfiles, getStreamUri) که در آن با استفاده از پاسخ رسیده و نوع درخواست، اطلاعات را از پاسخ استخراج کرده یا درخواست‌های بعدی را ارسال کرد.

بدین ترتیب پس از تعریف دستگاه با استفاده از ورودی‌های کاربر، Listener آن را فرخوانده و درخواست getServices را به دوربین ارسال می‌کنیم. فرستادن این درخواست قبل از گرفتن اطلاعات دستگاه الزامی نیست ولی از آنجایی که این دستور مسیرهای مختلف را که برای هر دوربین متفاوت است بر اساس وب سرویسی که به آن درخواست می‌دهید برمی‌گرداند استفاده از آن توصیه می‌شود. برای مثال دستور

getProfiles در یک دوربین Dahua مسیر /onvif/media2_service و در یک دوربین Uniview مسیر /onvif/media2 را خواهد داشت.

برای مشاهده‌ی تصاویر زنده‌ی دوربین نیاز داریم که پروفایل‌های مختلف دوربین را استخراج کرده و از درون آنها آدرس پخش RTSP را بازیابی کنیم. این روش این مزیت را دارد که با اینکه آدرس پخش هر برند دوربین متفاوت است چون آدرس از اطلاعات خود دوربین بازیابی می‌شود ما نیاز به ذخیره آدرس پخش دوربین در کد یا تغییر آن نداریم.

برای گرفتن آدرس پس از ارسال getServices و دریافت پاسخ آن در requestPerformed() در صورتی که پاسخ با موفقیت دریافت شده و با توجه به نوع درخواست ارسالی به ترتیب درخواست‌های getDeviceInformation، getProfiles و getStreamUri را به همین منوال در هر مرحله ارسال می‌کنیم و در آخر ئر صورت موفقیت آمیز بودن درخواست‌ها و گرفتن اطلاعات از پاسخ آنها با زدن دکمه‌ی Connect to Camera اطلاعات دوربین مورد نظر را نمایش داده و با زدن دکمه‌ی Play Stream آدرس پخش RTSP بازیابی شده را به Stram Activity برای نمایش پاس می‌دهیم.



شکل ۴: نحوه‌ی اجرا Main Activity

Stream Activity:

از آنجایی که آدرس پخش از پروتکل RTSP تبعیت می‌کند و VideoView اندروید ممکن است در پخش آن مشکلاتی داشته باشد (زیرا همه‌ی codecها را پشتیبانی نمی‌کند که مناسب دوربین‌های ONVIF که codecهای مختلفی دارند نیست) می‌توان از کتابخانه‌ی VLC که مناسب‌تر است استفاده کرد. همچنین از آنجایی که نحوه‌ی اجازه‌ی دسترسی به قسمت‌های حساس و شخصی تلفن کاربر پس از اندروید ۶ به گونه‌ای تغییر کرده که باید علاوه بر کسب اجازه در هنگام نصب در هنگام اجرای برنامه نیز از کاربر اجازه‌ی دسترسی گرفته شود برای استفاده از قابلیت ارسال پیام که یکی از این دسترسی‌های حساس است به کمک کتابخانه‌ای مراحل کسب اجازه را کوتاه‌تر انجام می‌دهیم. برای ارتباط با وب سرویس و ارسال درخواست فرمان‌های حرکتی از آن طریق نیز به کتابخانه‌ی Volley نیاز داریم پس قسمت‌های زیر را به build.gradle اضافه می‌کنیم:

```
allprojects {
  repositories {
    maven { url 'https://jitpack.io' }
  }
}
```

و در dependencies موارد زیر را می‌افزاییم:

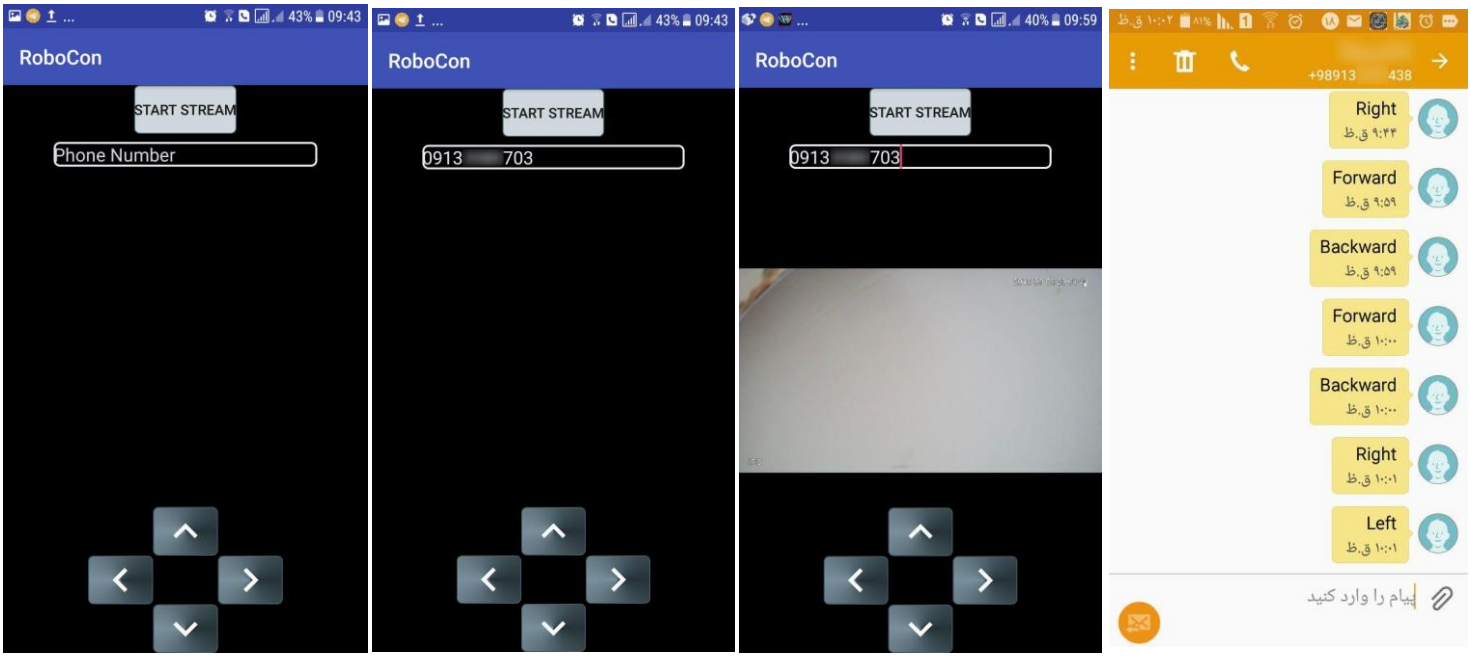
```
implementation 'com.github.pedroSG94.vlc-example-streamplayer:pedrovlc:2.5.14'
implementation 'gun0912.ted:tedpermission-rx2:2.2.2'
implementation 'com.android.volley:volley:1.0.0'
```

پس از زدن دکمه‌ی Start Stream در Main Activity و شروع Stram Activity ابتدا اجازه‌ی استفاده از قابلیت ارسال پیام از تلفن کاربر را از وی می‌گیریم. در صورتی که کاربر این اجازه را بدهد کدهای وابسته به این اجازه (در اینجا کدهای ارسال پیام های حاوی دستورات حرکتی به شماره‌ی مورد نظر از طریق دکمه‌ها) را در تابع onPermissionGranted() نوشته در غیر این صورت در تابع onPermissionDenied() به کاربر اطلاع می‌دهیم که استفاده از این سرویس نیاز به این اجازه داشته و تا داده نشدن آن امکان ارسال پیام نیست.

برای ارسال پیام از تابع sendSMS() استفاده می‌کنیم که شماره‌ی مورد نظر (که از کاربر گرفته می‌شود) و متن پیام را دریافت کرده و با استفاده از تابع sendTextMessage() از کلاس SmsManager پیام ما را از درون برنامه ارسال می‌کند. این تابع در صورت داده شدن اجازه ارسال پیام درون setOnClickListener هر کدام از دکمه‌های حرکتی فرمان مربوط به همان دکمه را به شماره‌ی مورد نظر کاربر ارسال می‌کند.

برای پخش تصاویر دوربین یک متغیر از نوع کلاس VlcVideoLibrary ایجاد کرده و آدرس پخش RTSP را که از activity قبل فرستاده‌ایم در صورت زدن دکمه‌ی Start Stream به آن می‌دهیم. این کلاس دو تابع onComplete() و onError() دارد که توسط کلاس VlcVideoLibrary به ترتیب در هنگامی که ویدیو در حال بارگیری است و یا خطایی رخ داده فراخوانده می‌شوند.

تابع move() ساختار دستورات مورد نیاز برای ارسال درخواست‌های رشته‌ای به وب سرویس مورد نظر برای ارسال فرمان‌های حرکتی از طریق اینترنت را دارد. بنابراین پس از اطلاع از نحوه‌ی ارسال درخواست‌ها به وب سرویس مذکور، پارامترها و ساختار درخواست و نوع پاسخ ارسالی از طرف آن (که همه از طرف ارائه دهنده‌ی وب سرویس مشخص می‌شود) می‌توان با تکمیل قسمت‌های مورد نیاز در بدنه تابع و تغییرات مورد نیاز از این روش نیز برای ارسال فرمان‌ها استفاده کرد.



شکل ۵: نحوه‌ی اجرا Stram Activity

نتیجه گیری:

پس از پیاده سازی ما موفق شدیم تصویر دوربین تحت شبکه که بر روی ربات نصب خواهد شد را دریافت کرده و از این مزیت برخوردار باشیم که با استفاده از پروتکل ONVIF امکان استفاده از دوربین‌های برند دیگر و یا تغییر اطلاعات احراز هویت دوربین خود را بدون اینکه نیاز به تغییر در کد برنامه باشد داشته باشیم. در ادامه می‌توان با استفاده از دکمه‌های کنترلی و در حین مشاهده تصویر دستورات را برای هدایت ربات به محل‌های مورد نظر از طریق اینترنت (در صورت راه‌اندازی سایر پارامترهای مورد نیاز) و یا از طریق پیام ارسال کرد.

مراجع:

1. ONVIF. (2018 Aug 15). Retrieved from https://en.wikipedia.org/wiki/ONVIF
2. Real Time Streaming Protocol. (2018 Jun 4). Retrieved from https://en.wikipedia.org/wiki/Real_Time_Streaming_Protocol
3. IP Camera (2018 Aug 20). Retrieved from https://en.wikipedia.org/wiki/IP_camera
4. Virin,R. (2018 Mar 8). ONVIF Camera Android Library Retrieved from https://github.com/rvi/ONVIFCameraAndroid
5. Park,S. (2018 Sep 4). TedPermission Android Library Retrieved from https://github.com/ParkSangGwon/TedPermission
6. SmsManager (2018 Jun 6). Reterieved from https://developer.android.com/reference/android/telephony/SmsManager