گزارش کار تمرین سوم

علیرضا سالمی - هومان چمنی - رضا قنبری - مهدی صالحی دزفولی

توضیحات کد:

بازی به طور کلی شامل دو Activity می‌شود. در Activity اول که نام آن MainActivity است برای هر mode از بازی (سنسور شتاب یا سنسور ژیروسکوپ) یک Button در نظر گرفته شده تا نوع بازی مشخص شود. در Activity دوم که نام آن GameActivity می‌باشد صفحه بازی شامل توپ و دکمه ریست ساخته شده و بازی شروع می‌شود(از طریق شروع لوپ اصلی بازی به نام gameLoop).عکس زیر نیز این مورد را نشان می‌دهد.

```
@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_game);
    randomSpeed_btn = (FloatingActionButton) findViewById(R.id.gyroscope_random_speed_btn);
    Intent intent = getIntent();
    sensorType = intent.getStringExtra( name: "sensor_type");
    randomSpeed_btn.setOnClickListener(v -> {
        gameLoop.pushBall();
    });
    gameView = (GameView) findViewById(R.id.game_view);
    SensorManager sensorManager = (SensorManager) getSystemService(Context.SENSOR_SERVICE);
    DisplayMetrics displayMetrics = new DisplayMetrics();
    getWindowManager().getDefaultDisplay().getMetrics(displayMetrics);
    Pair<Integer,Integer> screen = new Pair<>(displayMetrics.widthPixels, displayMetrics.heightPixels);
    gameLoop = new GameLoop(gameView,sensorManager,sensorType,  dt: 1,screen);
    gameLoop.start();
    getWindow().setFlags(WindowManager.LayoutParams.FLAG_FULLSCREEN,
            WindowManager.LayoutParams.FLAG_FULLSCREEN);
    getWindow().addFlags(WindowManager.LayoutParams.FLAG_KEEP_SCREEN_ON);
}
```

یک کلاس Ball نیز داریم که وظیفه نگه‌داشتن مختصات توپ را داشته و مواردی مانند نگه‌داشتن شعاع و آپدیت شدن مختصات به نقطه‌ای جدید نیز درون آن است.

کلاس بعدی کلاس MovementRecognizer می‌باشد که به تشخیص نوع حرکت می‌پردازد. در این کلاس مشخص می‌شود که حرکت چه نوعی است (منظور از نوع این است که سقوط آزاد است یا برخورد و یا لغزش) و در صورتی که حرکت از نوع برخورد یا لغزش باشد، دیواره‌ای که در آن این حرکت رخ داده شده هم مشخص می‌شود. از این کلاس برای محاسبه‌ی مختصات بعدی توپ در کلاس LocationCalculator استفاده می‌شود.

```
public class MovementRecognizer {

    private int xLength;
    private int yLength;
    private int radius;
    private int axis;

    public MovementRecognizer(int xLength, int yLength, int radius) {
        this.xLength = xLength;
        this.yLength = yLength;
        this.radius = radius;
        this.axis = 0;
    }
```

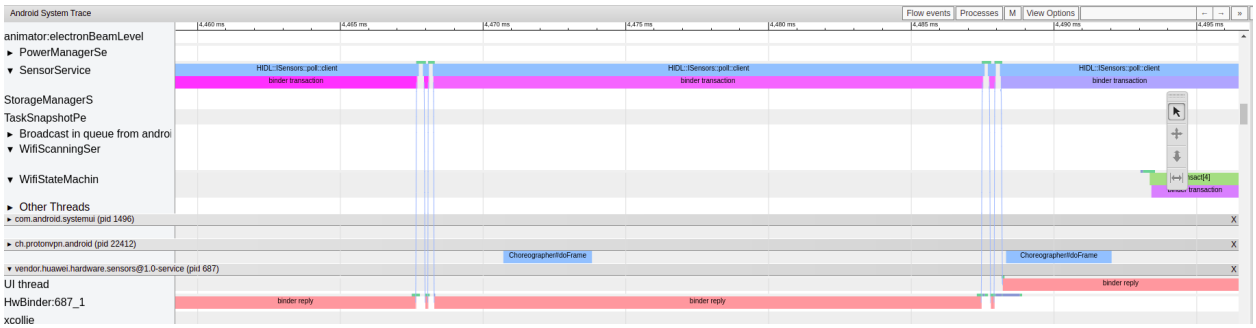
کلاس اصلی محاسبه‌کننده محل بعدی توپ، کلاس LocationCalculator است که قوانین فیزیکی مربوط به هر یک انواع حرکت را پیاده‌سازی می‌کند. این کلاس با استفاده از شتاب به محاسبه‌ی مکان و سرعت توپ می‌پردازد و برای دریافت شتاب نیز از کلاس‌های GravityLocationCalculator و GyroscopeLocationCalculator استفاده می‌کند که وظیفه‌ی خواندن مقدار سنسور و محاسبه‌ی مقدار شتاب در جهت محورهای مختصات را بر عهده دارند.

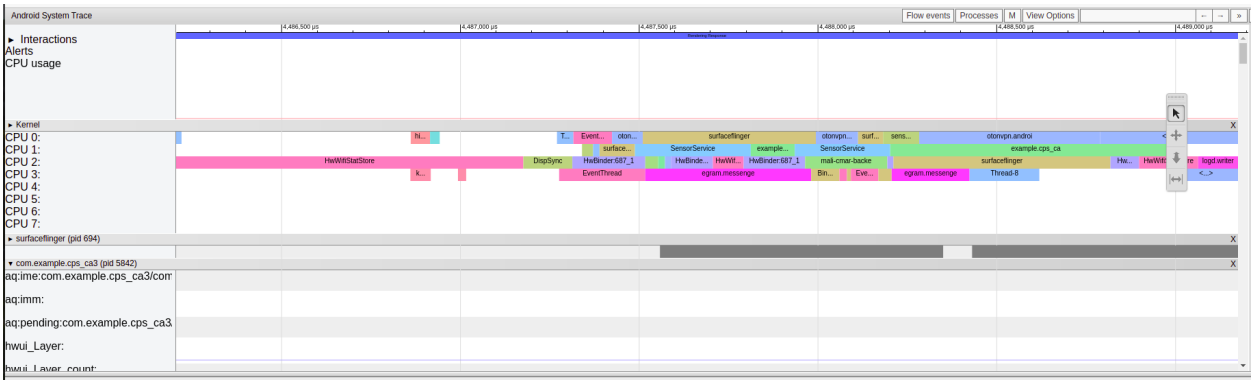
```java
public abstract class LocationCalculator {

    protected SensorManager sensorManager;
    protected Sensor sensor;
    protected State previous;
    protected State current;
    protected MovementRecognizer recognizer;
    protected double deltaT;
    private final double m = 0.01;

    public LocationCalculator(SensorManager sensorManager, Sensor sensor, State start, MovementRecognizer recognizer) {
        this.sensorManager = sensorManager;
        this.sensor = sensor;
        this.previous = start;
        this.current = start;
        this.recognizer = recognizer;
    }
```

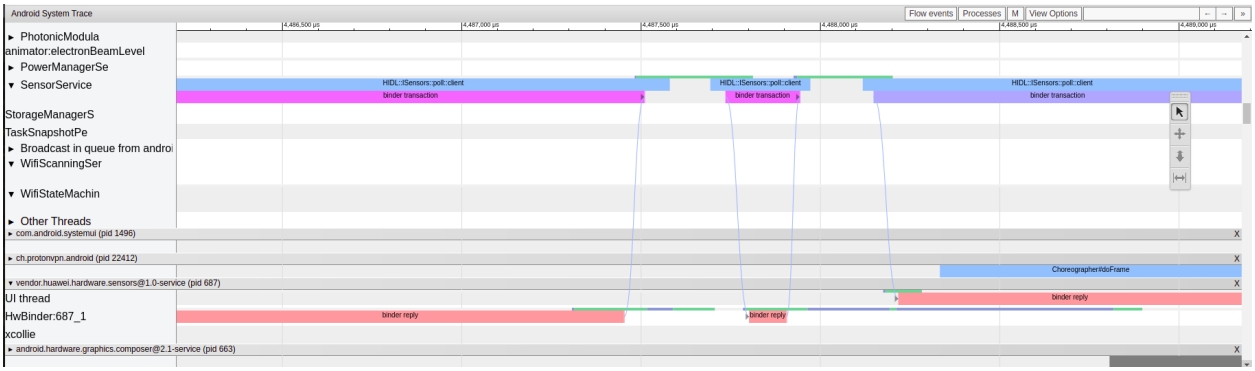
سوالات پروژه:

۱: از وقتی که درخواست خواندن داده به سنسور داده شده تا گرفتن داده چه اتفاقاتی در سطح سیستم‌عامل افتاده است؟ توضیح خود را با خروجی systrace توضیح داده و توجیه کنید.

ابتدا تصاویر زیر که مربوط به لحظه خواندن اطلاعات از سنسور است را مشاهده کنید:





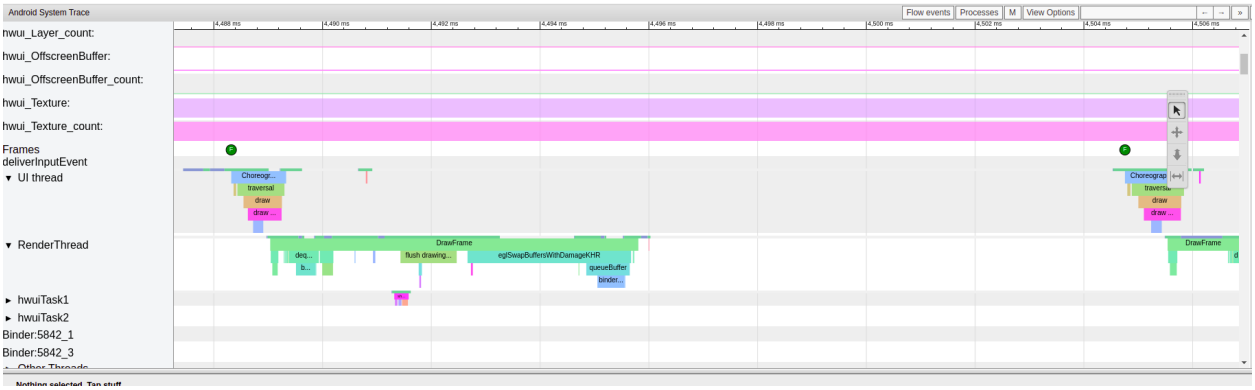


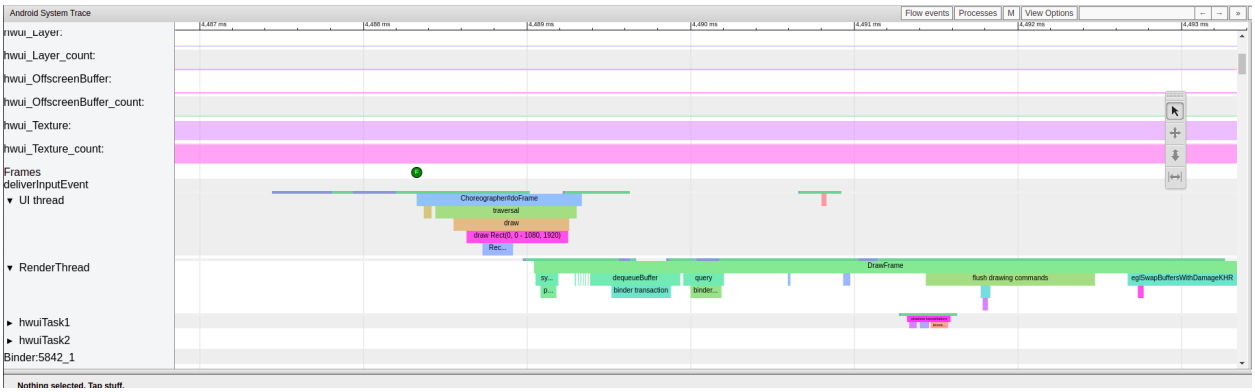
در تصویر اول مشخص است که فرایند خواندن داده از سنسورها در سطح سیستم عامل به صورت دوره‌ای و در حدود هر ۱۸ میلی‌ثانیه انجام می‌گیرد. اما این مورد در سطح سخت افزار به صورت دائمی است (با بازه فواصل آن بسیار کم است که در این نمودار در اسکیل میلی‌ثانیه قابل مشاهده نیست) مگر در زمانی که درخواست داده از سطح بالای سیستم عامل آمده باشد. در این بازه اگر از نظر زمانی نمودار اول و دوم را باهم مقایسه کنید می‌بینید که در لحظه‌ای که دستور مربوط به Sensor Service در سطح ریسه‌های cpu اجرا می‌شود (در تصویر دوم)، Sensor Service عملیات خواندن داده را برای لحظه‌ای متوقف می‌کند و اطلاعات را به سطح بالاتر تحویل می‌دهد و پس از آن دوباره می‌بینم در ریسه مربوط به آن HDL:sensor:poll:Client اجرا می‌شود که یعنی دارد از سنسور داده را دریافت می‌کند ولی لزوما به listener ها تحویل نمی‌دهد. همچنین دقت کنید که برای انتقال داده‌ها بین سخت افزار و sensor service نیز یک hardware binder وجود دارد که وظیفه انتقال این داده‌ها را بر عهده دارد. این اتفاقی است که بین سخت افزار و sensor service رخ می‌دهد. اما برای مشاهده اتفاقات در سطح سیستم عامل

باید نمودار دوم را مشاهده نمایید. در اینجا اتفاقات متفاوتی وجود دارد که لزوما همگی نیز مربوط به sensor service نیستند ولی sensor service هم در بین آن‌ها وجود دارد.

۲: چه مدت زمان طول می‌کشد تا مکان جدید گوی بر اساس مقداری که از سنسور خوانده شده روی صفحه نمایش ظاهر شود؟

برای بررسی این موضوع دو تصویر زیر را در نظر بگیرید:





در تصویر اول مشاهده می‌کنید که مدت زمان فاصله بین دو فراخوانی draw Frame در حدود ۱۷ میلی‌ثانیه است که چون صفحه نمایش تلفنی که بر روی آن تست کردیم ۶۰ هرتز است، این عدد کاملا با آن یعنی فرکانس آپدیت صفحه نمایش مطابقت دارد. ما نیز نرخ آپدیت صفحه را ۱۶ میلی ثانیه قرار داده‌ایم زیرا کمتر از ۱۶ میلی ثانیه بی‌معنا است. همچنین دقت کنید هر دفعه آپدیت کردن صفحه حدود ۱ میلی ثانیه زمان می‌برد.

۳: بنظر شما بهترین دوره تناوب براي خواندن مقادیر سنسورها و محاسبه مکان گوي چه مقدار است؟

دوره تناوب می‌بایست یک کمینه و بیشینه داشته باشد. کمینه برای این باید باشد زیرا به هر حال محاسباتی برای مشخص کردن مختصات بعدی گوی و شرایط محیط می‌بایست انجام شود و حالا هر چقدر هم سرعت محاسبات بالا باشد بازهم زمانی لازم است و اگر دوره تناوب کمتر این زمان باشد آنگاه این امکان به‌وجود خواهد آمد که تصویر خروجی که دریافت می‌کنیم نمود درست نباشد.این دوره تناوب یک بیشینه نیز باید داشته باشد زیرا اگر دوره تناوب خیلی زیاد باشد این ریسک وجود دارد که برخی اتفاقاتی که در سنسورها حس می‌شوند از دست بروند. یعنی مثلا تغییری سریع در زاویه داشته باشیم و چون دوره تناوب بالاست قادر به تشخیص دادن آن نباشیم. البته دوره تناوب مناسب به ویژگی‌های دستگاهی که با آن کار می‌کنیم داشته که این ویژگی‌ها شامل قدرت پردازنده و حافظه و غیره می‌باشند.چون اغلب دستگاه‌ها نرخ بروزرسانی صفحه آنان حدود 60Hz می‌باشد لذا 16ms عدد نسبتا مناسبی به نظر می‌رسد. البته بازهم موارد دیگر را باید در نظر داشته باشیم و این صرفا یک تخمین از روی میانگین است.

۴: بنظر شما اگر از Android NDK بجاي Android SDK استفاده مي شد، بازي شما چه مزايا و معايبی داشت؟

لغت SDK مخفف عبارت Software Development Kit می‌باشد. این کتابخانه برای اجرای خود از ماشین مجازی استفاده کرده و پایه آن نیز جاوا می‌باشد و صرفا برای برنامه‌نویسی برنامه‌های دستگاه android کاربرد دارد. یکی از بدی‌های آن در مقابل NDK همین ماشین مجازی بوده که اندکی سرعت را کاهش می‌دهد. در مقابل NDK این کتابخانه شامل ماژول‌ها و کتابخانه‌های گسترده‌تری بوده و به دلیل همین گستردگی SDK خود شامل همه API هایی که اغلب مورد نیاز هستند می‌شود.

در نقطه مقابل NDK قرار دارد که مخفف عبارت Native Development Kit می‌باشد. این کتابخانه اندکی از SDK پیچیده‌تر بوده و برای اجرای خود نیازی به ماشین مجازی نداشته و از طریق interface خود جاوا منابع مورد نیاز خود را تامین کرده لذا می‌توان سرعت اجرای بالاتری نسبت به SDK داشته باشد. پایه این کیت زبان C (و

منطقا Cpp) می‌باشد و به همین دلیل علاوه بر خود android قابلیت استفاده از برنامه‌های خروجی روی ویندوز و حتی IOS نیز وجود دارد. بر خلاف SDK که درون خود API های لازم را داشت در NDK این قابلیت وجود نداشته و بعضا با کاهش کارایی مواجه هستیم و شاید حتی پشتیبانی از همه Functionality های مورد تصور قابل انجام نباشد.

۵:در مورد سنسورهاي hardware-based و software-based تحقیق نمایید و هر یك را تشریح نمایید. هر کدام از سنسورهاي مورد استفاده در این تمرین در کدام دسته قرار مي‌گیرند.

سنسورهای software-based (به اختصار sb) که به آنان سنسور مجازی نیز می‌گویند آن دسته از سنسورها هستند که برای کار کردن از سنسورهای hardware-based (به اختصار hb) استفاده می‌کنند. یعنی sb ها رفتارهایی که hb ها در حالت عادی انجام می‌دهند را شبیه‌سازی می‌کنند. همچین یک سری سنسور hb نیز وجود دارند که وظیفه تامین منابع و اطلاعات لازم برای کار کردن sb ها را دارند. سنسور gravity که در پروژه استفاده شده است هم نوع sb و هم نوع hb دارد. در دسته دیگر سنسورهای hb وجود دارند که به طور فیزیکی و مثلا یک تراشه وجود دارند و بر خلاف sb ها دیتا مورد نیاز خود را با اندازه‌گیری‌هایی که خود انجام می‌دهند جمع‌آوری می‌کنند. سنسور gyroscope که در این پروژه استفاده شده است از نوع hb می‌باشد.

۶:تفاوت سنسورهاي Gravity و Gyroscope را تشریح نمایید.این تفاوت ها چه تاثیری بر محاسبات شما داشته است؟

سنسور gyroscope همان‌طور که از نامش پیداست تغییراتی که در زاویه دستگاه به‌وجود می‌آید را اندازه‌گیری می‌کند. برای انجام این کار زوایایی که دستگاه در حالت قبلی با سه محور داشته نگه‌داری می‌شوند و زوایای حالت جدید طبق تغییرات انجام شده و مقادیر قبلی محاسبه می‌شود. در طرف دیگر سنسور gravity وجود دارد که شتاب جاذبه

را در راستای سه محور اصلی اندازه‌گیری می‌کند. چون در اغلب مواقع دستگاه به طور کاملا عمود بر شتاب گرانش اصلی نبوده شتاب در راستای هر سه محور طبق زاویه دستگاه باید اندازه‌گیری شده و گزارش شود.چون که سنسور gravity خود نیروی جاذبه اعمال شده در سه جهت را به ما می‌دهد برای محاسبه شتاب در جهات مختلف از همان خروجی سنسور gravity در راستاهای مختلف استفاده می‌کنیم اما در استفاده از سنسور gyroscope چون زاویه دستگاه با سه محور داده می‌شود ابتدا می‌بایست زاویه را تبدیل به شتاب کرده (با استفاده از روابط مثلثاتی و عدد شتاب که ۹.۸ در نظر گرفته شد) و بعد محاسبات را ادامه داد.

۷:-در صورتي که بازي در حالتي شروع شود که گوشي روي سطح شيبدار قرار داشته باشد، چه اتفاقی می افتد؟ در این حالت آیا تفاوتي ميان استفاده از سنسور Gyroscope و Gravity وجود دارد یا خیر؟ توضیح دهید

در صورتی که در این شرایط از سنسور ژیروسکوپ استفاده کنیم بازی خراب می‌شود. دلیل این مورد این است که شیب که دستگاه در حالت اولیه در آن قرار دارد (همان سطح شیبدار) هنوز مشخص نشده است و ژیروسکوپ نیز همان‌طور که در بالا توضیح داده شد با استفاده از مقدار اولیه و تغییر انجام شده زوایای جدید را محاسبه می‌کند پس عملکرد آن دچار مشکل شده و فیزیک درستی از بازی مشاهده نخواهد شد. در صورتی که از سنسور جاذبه استفاده کنیم این مشکل وجود نخواهد داشت زیرا طبق توضیحات داده شده این سنسور کاری به حالتی که دستگاه در ابتدا در آن قرار دارد نداشته و صرفا در هر لحظه مقدار شتاب را در راستای سه محور گزارش می‌دهد و با همین موارد می‌توان فیزیک بازی را به درستی محاسبه کرده و جلو رفت.