实 训 报 告

实训名称：基于飞腾平台的远距离探测系统

姓　　名：	梅骏逸	学　　号：	2111876
专　　业：	信息安全-法学	年　　级：	2021 级
所属学院：	网络空间安全学院	实训日期：	2023.06.27-06.30

飞腾信息技术有限公司

《实训报告》填写说明

1．学生完成课程任务所要求的实训后，均须提交实训报告。

2．实训报告提交电子版文档。

3．实训报告内容编排应符合以下要求：

（1）页面设置采用 A4（21cm×29.7cm）白底黑字。上下各侧的页边距均为 2.5cm；左右各侧的页边距均为 3cm；缺省文档网格：字号为小 4 号，中文为宋体，英文和阿拉伯数字为 Times New Roman，每页 30 行，每行 36 字；页眉距边界为 1cm，页脚距边界为 1.5cm，页码置于页脚、居右，采用小 5 号阿拉伯数字从 1 开始连续编排，封面不编页码。

（2）报告正文最多可设四级标题，字体均为黑体，第一级标题字号为 4 号，其余各级标题为小 4 号；标题序号第一级用“一、”、“二、”……，第二级用“（一）”、“（二）”……，第三级用“1.”、“2.”……，第四级用“（1）”、“（2）”……，分别按序连续编排。

（3）正文插图、表格中的文字字号均为 5 号。

一、实训目的

- ✧ 能够熟练操作飞腾教育开发板。
- ✧ 能够在飞腾教育开发板上完成 C 工程开发。
- ✧ 能够灵活运用飞腾平台 VSIPL 函数库实现信号处理算法。
- ✧ 能够锻炼学生举一反三的能力。
- ✧ 能够锻炼学生的自学与探究能力。

二、实训内容

基于飞腾教育开发板构建一个远距离探测系统；

三、已知条件

- ✧ 雷达发射的脉冲宽度为 7us
- ✧ 线性调频信号的下限截止频率为 222MHz，带宽为 6MHz
- ✧ 各个脉冲的初相位均为 0
- ✧ 信号在模拟接收电路中会通过带通滤波器（225MHz±3 MHz）
- ✧ 采样率为 20MHz
- ✧ 浮点数均是 single

四、实训要求

Phytium飞腾

课后作业

1、自行封装汉明窗接口函数，匹配滤波器（复模板）的窗函数改成汉明窗；
vsip_vview_f* vsip_vcreate_hamming_f(vsip_length N);

2、利用余弦函数构造一个雷达回波脉冲（实信号），使得它是由两个相距600米的静止物体反射叠加形成，并且在回波信号上叠加高斯白噪声，使得其信噪比为0dB；

3、设计一个不超过10阶的希尔伯特滤波器，通过时域滤波，将雷达回波由实信号转换成与之对应的复信号，然后再做脉冲压缩；

4、在不清楚目标数量和间距的情况下，通过检测手段，反向证明目标个数为2，相距600米；

芯科技 兴飞腾

五、原理分析

1、建立物理模型、描述系统工作原理。

远距离探测系统首先接收一个雷达回收脉冲实信号，将这一信号通过希尔伯特滤波得到对应的复信号。之后，将此前发送出去的信号作为参考信号，共轭翻转之后加上汉明窗，汉明窗可以用如下公式表示：

$$w(n) = a_0 - (1 - a_0)\cos\left(\frac{2\pi n}{N-1}\right), \quad 0 \le n \le N-1$$

其中 a_0 的值设置为 0.53836 即为汉明窗。

得到的参考信号和经过希尔伯特滤波之后得到的复信号经过傅里叶变换之后在频域上相乘再经过逆傅里叶变换即可得到最终脉冲压缩结果在时域上的表示。由于原始信号存在噪声，所以需要对脉冲压缩的结果进一步筛选，从而定位目标发送信号的时间以及目标之间的距离。

2、如何构造雷达回波信号。

线性调频信号的相位可以表示为：

$$\varphi(t) = 2\pi f_{\text{low}} t + \frac{\pi \text{BW}}{\tau} t^2$$

线性调频的复信号则可以用欧拉公式表示为：

$$s(t) = e^{i\varphi(t)}$$

由于雷达回波得到的是实信号，所以我们只需要取复信号的实部即可，即应用余弦函数

$$s_{\text{real}}(t) = \cos\left(2\pi f_{\text{low}} t + \frac{\pi \text{BW}}{\tau} t^2\right)$$

设两个物体相聚为 d，则两个物体的回波相差的时间为

$$\Delta t = \frac{2d}{c}$$

其中 c 为光速， d 为两个物体的距离。假设接收到第一个物体反射信号的时间为 t_0，则雷达接收到的两个物体的信号分别为

$$\begin{aligned} s_1(t) &= \cos\left(2\pi f_{\text{low}}(t - t_0) + \frac{\pi \text{BW}}{\tau}(t - t_0)^2\right) \\ s_2(t) &= \cos\left(2\pi f_{\text{low}}(t - t_0 - \Delta t) + \frac{\pi \text{BW}}{\tau}(t - t_0 - \Delta t)^2\right) \end{aligned}$$

最终叠加得到的信号为

$$s(t) = s_1(t) + s_2(t)$$

3、如何叠加高斯白噪声使得回波信号信噪比为 0。

信噪比的定义如下（单位为分贝）：

$$\text{SNR} = 10\log_{10}\frac{P_{\text{signal}}}{P_{\text{noise}}}$$

对于离散的采样，信噪比可以表示为：

$$\text{SNR} = 10\log_{10}\frac{\sum_{i=0}^{N-1} x_i^2}{\sum_{i=0}^{N-1} n_i^2}$$

其中 x_i 为信号，n_i 为噪声，且振幅满足分布：

$$n \sim \mathcal{N}\left(0, \frac{\sum x_i^2}{10^{\left(\frac{\text{SNR}}{10}\right)}N}\right)$$

依照这一分布生成噪声信号并叠加在原始信号上即可得到信噪比为 0dB 的回波信号。

4、设计希尔伯特滤波器的全过程。

定义符号函数：

$$\text{sgn}(x) = \begin{cases} 1, & x > 0 \\ 0, & x = 0 \\ -1, & x < 0 \end{cases}$$

对于时域信号 $x(t)$，设其频域表示为 $X(\Omega)$，则在频域下的希尔伯特变换可以表示为：

$$\hat{X}(\Omega) = [-j\text{sgn}(\Omega)]X(\Omega)$$

假设希尔伯特变换之后信号的时域表示为 $\hat{x}(t)$，则希尔伯特滤波结果的时域表示为：

$$s(t) = x(t) + j\hat{x}(t)$$

其频域表示为：

$$\begin{aligned} S(\Omega) &= X(\Omega) + j\hat{X}(\Omega) \\ &= X(\Omega) + j[-j\text{sgn}(\Omega)]X(\Omega) \\ &= [1 + \text{sgn}(\Omega)]X(\Omega) \end{aligned}$$

则希尔伯特滤波器的频域表示为：

$$H(\Omega) = 1 + \text{sgn}(\Omega)$$

表示为时域下的卷积运算：

$$s(t) = h(t) * x(t)$$

其中 $h(t)$ 为希尔伯特滤波器 $H(\Omega)$ 的时域表示，通过逆傅里叶变换可以得到如下表达式：

$$\begin{aligned} h(t) &= \mathcal{F}^{-1}\{H(\Omega)\} \\ &= \mathcal{F}^{-1}\{1 + \mathrm{sgn}(\Omega)\} \\ &= \delta(t) + \frac{1}{\pi t} \end{aligned}$$

但是由于所采用的是离散时间，所以需要对 $H(\Omega)$ 做离散时间傅里叶逆变换，得到 $h(t)$ 的离散表示：

$$\begin{aligned} h(n) &= \frac{1}{2\pi}\int_{-\pi}^{\pi}[1 + \mathrm{sgn}(\omega)]\, e^{j\omega n}\mathrm{d}\omega \\ &= \frac{1}{2\pi}\int_{0}^{\pi} 2\, e^{j\omega n}\mathrm{d}\omega = \frac{1}{\pi}\int_{0}^{\pi} e^{j\omega n}\,\mathrm{d}\omega \\ &= \frac{1}{\pi}\frac{e^{j\pi n} - 1}{jn} = \frac{\cos\pi n - 1}{j\pi n} = \frac{j(1 - \cos\pi n)}{\pi n} \\ &= \begin{cases} 1, & n = 0 \\ \dfrac{2j}{\pi n}, & n \text{ 为奇数} \\ 0, & n \text{ 为偶数} \end{cases} \end{aligned}$$

应用这一公式，求取十阶希尔伯特滤波器的系数，并且加上 0 处的 1 即可得到整个滤波器在时域下的系数，如前推导，将 $h(n)$ 与原实信号在时域下进行卷积可得到对应的复信号。

5、在不清楚目标数量和距离的情况下，利用通用化的检测手段，反向证明目标个数为 2，相距 600m。

根据一定的阈值对脉冲压缩之后所得到的信号进行筛选，之后线性遍历信号中离散的点，找到所有的局部最大点（绝对值同时大于左右两边的值），以及这个点所对应的下标，从而计算对应的时间。依据对应的下标和采样率得到时间，从而得到目标的距离。首先根据采样率 f_{sample} 得到采样的间隔

$$\Delta t_{\mathrm{sample}} = \frac{1}{f_{\mathrm{sample}}}$$

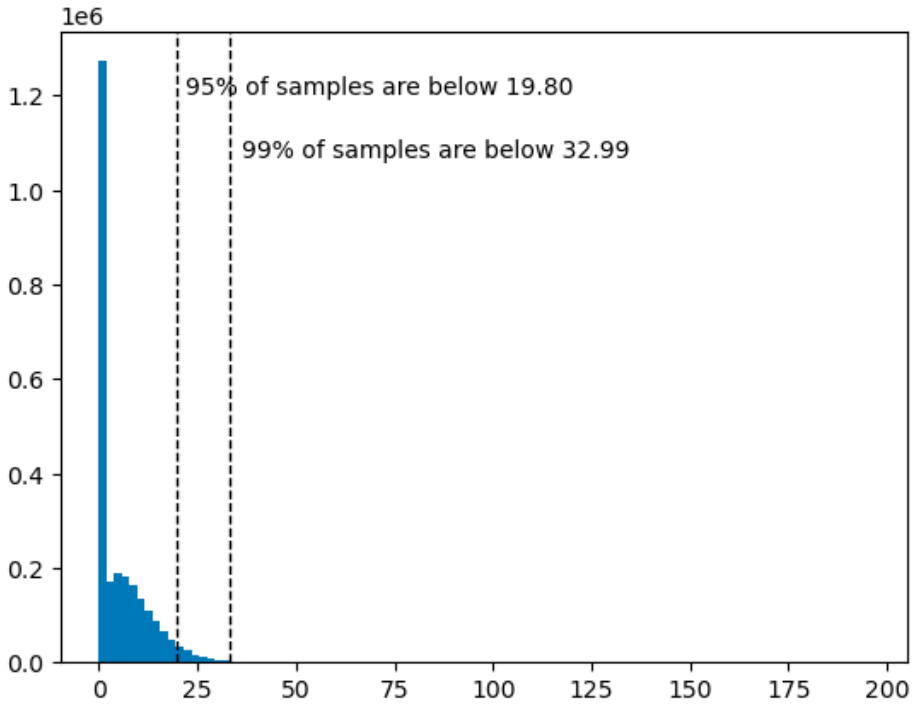
假设检测到了两个相邻的峰值，且二者的下标相差 n，则二者的时间间隔为

$$\Delta t = n\Delta t_{\mathrm{sample}}$$

最后根据光速计算得到距离：

$$d = \frac{c\Delta t}{2}$$

对于脉冲压缩之后的信号进行筛选的阈值通过随机和采样的方式进行获取，对于线性调频信号，随机生成五千组对应的噪声并且统计其功率（绝对值）的大小的分布情况，根据一定的概率取得统计结果的阈值，统计采样得到的结果如下：

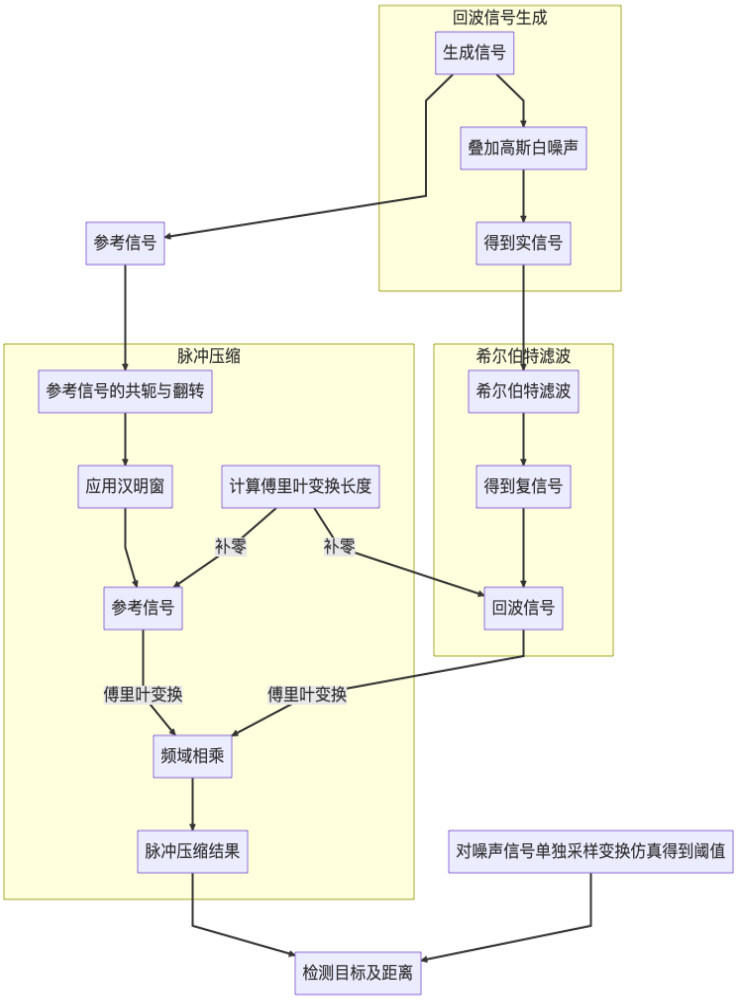


代码实现过程中取信号绝对值的阈值为 30，绝对值大于这个值的点被保留，剩余则置为 0，从而实现从脉冲压缩的信号中提取峰值。在实验中发现这一方法能够实现对目标的检测功能。

六、软件详细设计

必须包含流程图与关键接口说明，且最终依靠软件求解目标个数与间距。

（一）流程图



（二）关键接口说明

希尔伯特滤波

```
/*
 * 内部接口：希尔伯特滤波
 * 参数：p_vector_src    -- 输入信号
 *      n_filter_length -- 滤波器长度
 *      p_vector_dst    -- 输出信号
 * 功能：对输入信号进行希尔伯特滤波
 */
void hilbert(vsip_vview_f *p_vector_src, vsip_scalar_i n_filter_length,
             vsip_cvview_f *p_vector_dst);
```

汉明窗

```
/*
 * 内部接口：生成汉明窗
 * 参数：p_vector_dst -- 输出信号
 * 功能：根据输出信号的长度生成汉明窗
 */
void vcreate_hamming_f(vsip_vview_f *p_vector_dst);
```

信号生成以及处理

```
/*
 * 内部接口：生成线性调频信号
```

```
 * 参数：f_tau           -- 脉冲宽度
 *      f_freq_sampling -- 采样频率
 *      f_freq_low      -- 起始频率
 *      f_band_width    -- 带宽
 *      p_vector_dst    -- 输出信号
 * 功能：生成线性调频信号（复信号）
 */
void generate_lfm_signal(vsip_scalar_f f_tau, vsip_scalar_f f_freq_sampling,
                         vsip_scalar_f f_freq_low, vsip_scalar_f f_band_width,
                         vsip_cvview_f *p_vector_dst);

/*
 * 内部接口：生成线性调频信号
 * 参数：f_tau           -- 脉冲宽度
 *      f_freq_sampling -- 采样频率
 *      f_freq_low      -- 起始频率
 *      f_band_width    -- 带宽
 *      p_vector_dst    -- 输出信号
 * 功能：生成线性调频信号（实信号）
 */
void generate_lfm_signal_real(vsip_scalar_f f_tau, vsip_scalar_f f_freq_sampling,
                              vsip_scalar_f f_freq_low, vsip_scalar_f f_band_width,
                              vsip_vview_f *p_vector_dst);

/*
 * 内部接口：生成雷达回波信号
 * 参数：f_tau           -- 脉冲宽度
 *      f_freq_sampling -- 采样频率
 *      f_freq_low      -- 起始频率
 *      f_band_width    -- 带宽
 *      f_distance      -- 两个物体之间的距离
 *      p_vector_dst    -- 输出信号
 * 功能：生成两个有一定距离的物体反射叠加得到的雷达回波信号
 */
void generate_radar_signal(vsip_scalar_f f_tau, vsip_scalar_f f_freq_sampling,
                           vsip_scalar_f f_freq_low, vsip_scalar_f f_band_width,
                           vsip_scalar_f f_distance, vsip_vview_f *p_vector_dst);

/*
 * 内部接口：生成雷达回波信号
 * 参数：p_vector_signal -- 输入信号
 *      f_snr           -- 目标信号信噪比
 *      p_vector_dst    -- 输出信号
 * 功能：生成可以叠加到原信号上的给定信噪比的高斯白噪声
 */
void generate_wgn_signal(vsip_vview_f *p_vector_signal, vsip_scalar_f f_snr,
                         vsip_vview_f *p_vector_dst);

/*
 * 内部接口：脉冲压缩
 * 参数：p_vector_signal_src -- 输入信号
 *      p_vector_signal_ref -- 参考信号
 *      p_vector_dst        -- 输出信号
 * 功能：使用给定的参考信号对输入信号进行脉冲压缩
 */
void pulse_compress(vsip_cvview_f *p_vector_signal_src, vsip_cvview_f *p_vector_signal_ref,
                    vsip_cvview_f *p_vector_dst);

/*
 * 内部接口：检测信号
 * 参数：p_vector_signal -- 脉冲压缩之后得到的信号
 *      f_threshold     -- 阈值
 *      p_vector_dst    -- 输出信号
 * 功能：对脉冲压缩之后的信号进行进一步检测，依据阈值进行筛选
 */
void detect_signal(vsip_cvview_f *p_vector_signal, vsip_scalar_f f_threshold,
                   vsip_cvview_f *p_vector_dst)
```

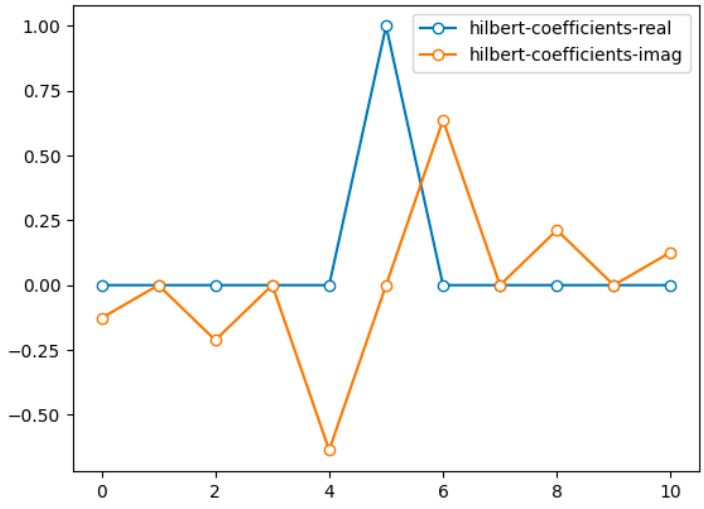
用于输出和调试的函数

```c
/*
 * 内部接口：输出实向量
 * 参数：p_vector -- 输入向量
 *      p_file   -- 输出文件
 * 功能：将实向量的数据输出到文件
 */
void vdump_f(vsip_vview_f *p_vector, FILE *p_file);

/*
 * 内部接口：输出复向量
 * 参数：p_vector -- 输入向量
 *      p_file   -- 输出文件
 * 功能：将复向量的数据输出到文件
 */
void cvdump_f(vsip_cvview_f *p_vector, FILE *p_file);

/*
 * 内部接口：实向量调试
 * 参数：p_vector -- 输入向量
 *      p_name   -- 输出文件名
 * 功能：将实向量的数据输出到指定文件名的文件
 */
void vdebug_f(vsip_vview_f *p_vector, char *p_name);

/*
 * 内部接口：复向量调试
 * 参数：p_vector -- 输入向量
 *      p_name   -- 输出文件名
 * 功能：将复向量的数据输出到指定文件名的文件
 */
void cvdebug_f(vsip_cvview_f *p_vector, char *p_name);

/*
 * 内部接口：复向量翻转
 * 参数：p_vector_src -- 输入向量
 *      p_vector_dst -- 输出向量
 * 功能：将输入向量的数据翻转后输出到输出向量
 */
void cvflip_f(vsip_cvview_f *p_vector_src, vsip_cvview_f *p_vector_dst);

/*
 * 内部接口：复向量填充
 * 参数：p_vector_src -- 输入向量
 *      p_vector_dst -- 输出向量
 * 功能：根据输出向量的长度对输入向量进行零填充得到输出
 */
void cvpad_f(vsip_cvview_f *p_vector_src, vsip_cvview_f *p_vector_dst);
```

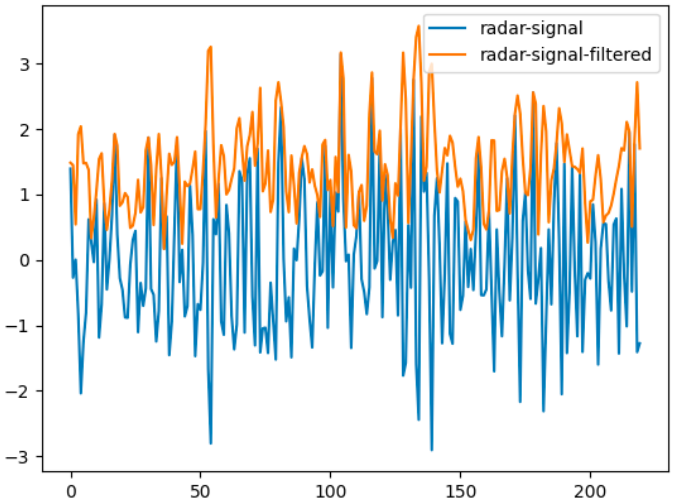
（三）最终结果

使用 VSIPL 库和 C 语言编写程序得到相应数据，之后使用 Python 进行绘图可以得到更加直观的结果如下。

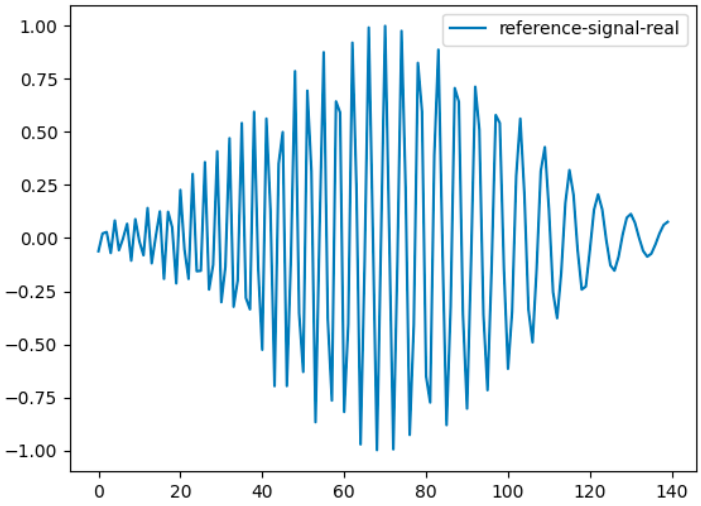
计算得到的希尔伯特滤波器系数的实部和虚部如下图（橙色为虚部，蓝色为实部）：



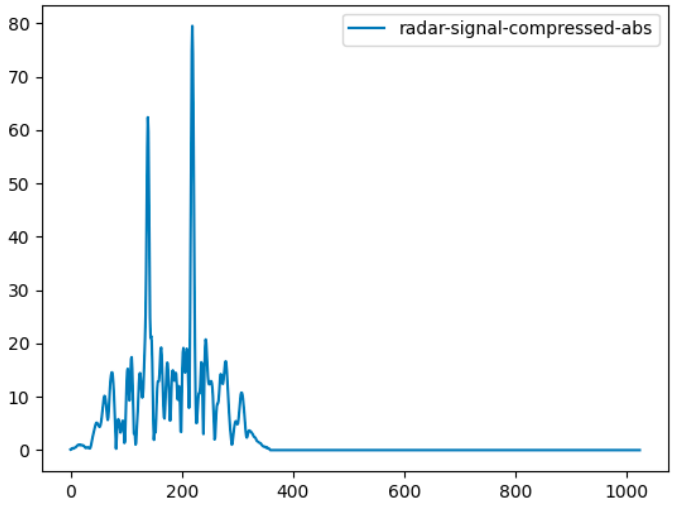
生成的加入高斯白噪声的雷达回波信号（实信号，蓝色）以及经过希尔伯特滤波之后得到复信号的功率大小（信号的幅值，橙色）如下图所示：



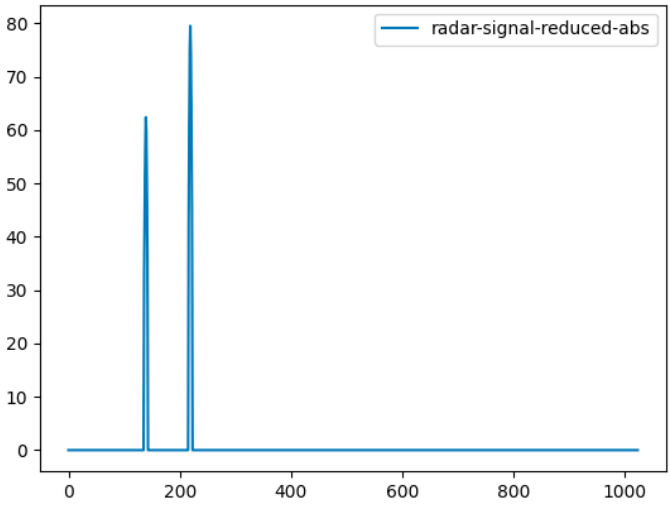
使用线性调频信号作为参考信号，共轭翻转之后加上汉明窗得到用于脉冲压缩的信号的实部如下图：



脉冲压缩得到的结果如下（幅值）：



使用阈值为 30 进行筛选得到结果：



在得到脉冲压缩信号中大于阈值的峰值之后，可以逐个遍历寻找向量中的局部最大值（同时大于左右的值的位置），并且计算对应的时间，通过时间差计算距离。最终从回波信号中检测到的目标以及距离如下：

```
kylin@10.136.181.197:22

kylin@kylin-phytiumft20004:~/phytium-vsipl-assignment$ make run
radar: n_fft_len = 1024
detect: time = 0.000007s
detect: time = 0.000011s, distance = 599.999939m
kylin@kylin-phytiumft20004:~/phytium-vsipl-assignment$
```

最终能够检测到两个目标，并且能够得到第二个目标与第一个目标之间的距离，距离的误差也在合理的范围内。

七、问题与建议

1、对实训过程中出现的问题进行描述、分析，提出解决思路和方法；无法解决的，要说明原因。

(1) 在实现希尔伯特滤波器的过程中发现当前的 VSIPL 版本没有复数向量进行卷积的操作函数。而由于希尔伯特滤波器中每一个系数为纯实数或纯虚数而不存在复合的复数，所以可以对实部和虚部分别进行卷积操作，最后合成复信号。

(2) 由于实现过程中主要使用 C 语言，最后生成的数据并不直观，故使用 Python 完成了一个简单的依照主程序计算出的数据生成对应图表的脚本。并且在

Makefile 中添加了 run 这一构建目标，通过 make run 命令进行运行可同时得到使用 VSIPL 的 C 程序生成的数据，以及依照这一系列数据绘制的图像内容。

(3) 试验中对脉冲压缩之后的信号进行筛选的阈值，是通过对噪声多次采样统计得到的，但是在信噪比未知的情况下需要让雷达系统多次直接接收噪声信号并进行统计从而确定。

2、记录实训心得体会，提出建议。

(1) 学习了 VSIPL 信号和图像处理库的有关接口以及使用。

(2) 了解了傅里叶变换、离散时间傅里叶变换、脉冲压缩、希尔伯特变换及滤波、高斯白噪声以及雷达算法的一些知识并且进行了一定的应用。

(3) 对信号在时域和频域上的操作以及相互之间的转换有了一定的理解。

八、源程序

附上完整的源程序（代码要有详细的注释）。

见工程源代码。