版权护卫

伴随着互联网和图像技术的快速发展，人们的生活与工作变得越来越方便，交流和娱乐也越来越多样化，但这同时也带来了不少安全隐患。回想过去十年中国互联网的野蛮发展，带来了大量的侵权和盗版问题，给著作人的利益带来了严重的损害，大量的数字内容行业遭受了深刻并且严重的破坏。幸运的是，现在版权问题已经获得了越来越多的重视，社会和国家也在促进版权保护的发展，各个公司更是视版权为财富。即便如此，由于数字技术的进一步发展，以及传统的可见水印本身的技术问题，使得可见水印技术不能跟上时代的要求，对版权的保护形同虚设。本平台将鲁棒不可见水印技术和互联网技术进行结合，为此类侵权盗版问题的解决提供了新的视角。虽然不可见水印技术目前仍然存在不足，因此没有大量的商业化，也不能彻底解决存在的安全问题，但是其提供了一种较为新颖的解决方案，并可以带来比可见水印更强的版权保护能力。

一、简介

本文产品的名称叫做“版权护卫”，以鲁棒的不可见水印技术为核心，构建了一套精简的版权保护平台，为用户提供最基本的版权保护能力的同时，也能保障图像的观赏性，甚至在图片在一定程度破坏的情况下，同样能够提取出水印。

产品主要提供了“可见水印”、“不可见水印”和“图片注册”的三大功能用以进行版权保护，这三种功能相互独立，每种功能都有其各自的应用场景。1).可见水印。该功能就是现在最场景的水印嵌入方案，嵌入一个肉眼可见的水印，水印内容是一串文本。2).不可见水印。该功能的水印内容同意是一串文本，但是嵌入后不可见，可以通过密钥将嵌入的水印文本提取出来。3).图片注册。该功能将原始图片进行注册，并返回一个注册图片，注册图片和原始图片之间的区别肉眼不可见。用户通过注册图片，可以追溯到该注册图片的注册人，并和注册人进行联系。

二、需求分析

2.1 现状分析

随着数字技术和通信技术的发展，图像成为目前互联网中最受欢迎的多媒体形式之一。2010 年以后，智能手机的大面积普及，也让越来越多的人通过手机拍照来记录生活点滴，再加上手机中各类滤镜的使用与后期手段，手机已经可以拍出不逊色于专业相机的效果，因此人们热于通过移动端的操作来进行图片的制作和分享。

发达的数字图像技术和便捷的分享途径，在给人们带来丰富乐趣和充实的图像内容的同时，也隐藏着多媒体安全问题。在互联网平台上发布自己创作的图片时，容易遭它人盗用，这将会极大侵害图片创作者的个人权益。据国家版权白皮书显示，2016 年全国版权产业整体产值突破 5600 亿元，在侵权盗版方面，网络图片侵权案件排名第二，占到了 24%。再比如，据北京市海淀区法院数据显示，2015 年该法院审理图片著作权案件 1000 多件，2016 年翻倍到 2000 件，2017 年中旬已经有 2800 多件，被告大部分是第三方平台微博、博客、APP 等平台等内容创作和分享平台。中国版权协会驻会副理事长王自强在 2017 年关于图片版权保护的研讨会上说到：“图片行业的地位很尴尬，最新的互联网版权报告竟然就没提图片版权的问题，而实际上，图片的侵权率在众多作品类型中位居第二。”

为了避免图片侵权发生时，图片作者的权益受到伤害，因此图片作者通常需要在发布前留下尽可能多的创作证据。在现阶段，作者在发布图像作品前，往往会通过加入可见水印的方式，留下最直观的证据，以达到最基本的保护版权的能力。水印中通常包含作者或图像本

身的相关信息，如图 2-1 中的照片加入了可见的水印信息“房天下 Fang.com”。

图 2-1 普通水印

可见水印的引入，改善了图片侵权的现状，也在一定程度上促进了大众的版权保护意识，但是可见水印也存在诸多的缺点亟待解决：

1).安全性过低。数字图像修复技术可以对破损图片进行一定程度的修复，水印本身就是一种可见的图片破损，因此水印极容易被包括数字图像修复技术在内的相关技术进行去除，使得不法分子可以从中还原出原图，进而满足自己的利益。对于一些放置于边边角角的水印而言，更是可以直接通过截图技术就能去掉水印。

2).对视觉的影响。为了保证安全性，可以采用将水印放置图片中央，甚至布满图片的形式，如图 2-2 所示。这样子的可见水印虽然提高了安全性，不易被截去，但是也对视觉效果带来了较大的破坏，影响了图片的观赏性，不利于图像放在互联网平台上传播。

图 2-2 强可见水印

3).真实性问题。为了提升安全性并且降低对视觉的影响，现在越来越多图像创作者喜

欢在图像中引入花式的水印，这些水印看起来较为真实，不容易被攻击者理解为水印，降低被去除的风险。如图 2-3 所示，碗上的“大阪”标识实际上为水印。虽然这样可以确实能起到效果，但是若被理解为水印，同意也容易被去除，并且这样的水印也容易让观赏者误会，不适合用于写实类的作品。

图 2-3 花式可见水印

为了解决可见水印存在的诸多问题，学术界在早期就已经提出了包含 LSB 在内的不可见水印技术，然而目前在不可见水印方面的小程序极少，在小程序搜索栏中搜索“水印”关键词，只能查询出寥寥几款可见水印小程序。不仅仅是小程序，在 APP、Web 和桌面应用中，都极少见到不可见水印平台的应用程序。虽然由于技术不够成熟，此类技术持续沉浸于实验室多年，但随着近几年鲁棒的不可见水印技术突飞猛进的发展，不可见水印越来越丰满和成熟，已经具备一定的商业使用价值。图 2-4 中分别显示了原始图像、水印图像和含水印的图像，很明显，从含水印图像中完全无法看出关于水印的信息。

(a).原始图像　(b).水印图像　(c).含水印图像

图 2-4 原始图像、水印图像与含水印图像

2.2 应用场景

2.2.1 版权保护

图片创作者发布不含水印的图片，虽然可以促进图片在互联网中的传播，但是却极易被盗用和篡改，因此如今很多创作者都嵌入可见水印并发布图片，水印中包含了创作者的相关信息，以便意图使用图片的人可以和图片创作者进行联系，并留下版权证据，但可见水印对图片带来的视觉影响，以及其较差的安全性，都使得其保护形同虚设。不可见水印技术可以最大程度降低对视觉的影响，其具有的鲁棒性也可以使得图片在受到一定程度破坏的情况下仍然能够提取水印，并且不可见水印本身也隐藏了图片“含水印”这一事实，避免引起水印攻击者的注意，提升安全性。本产品提供了“不可见水印”功能，提升图片安全性，并且促进图片在互联网中的传播。产品的“不可见水印”支持用户上传原始图片，并输入不可见水印的文本内容，并选填密钥和图片标题，小程序将会把必要数据上传到服务器，并由服务器

嵌入不可见水印，最后将含水印图像返回给小程序。希望使用图片的普通用户，可以在本平台上输入相关密钥提取出不可见水印。需要注意的是，在嵌入不可见水印时，图片创作人若是输入了密钥，这一行为有两个含义，一个是为了进行盗版追踪，另一个是意味着图片制作人禁止网络上的他人使用自己的图片，因为除了图片创作者，没有人知道密钥，也就无法提取作者的相关信息并和作者取得联系。图片创作人若是没有输入密钥，意味着每个人都可以提取出不可见水印的信息，水印中往往会有创作人留下的相关信息，意图使用图片的他人便可以和图片制作人取得联系。

我们在进行自己的创意设计时，通常会在网上查阅相关的图片，并引用网上查找到的图片，这些图片往往没有图片创作者的信息，若是在引用时稍不注意，极有可能导致侵权行为。另外一个方面，网络上存在着大量的虚假信息，有人为了谋取一己私利谎称是图片的作者。再一个方面，直接采用可见水印或是不可见水印的功能，将会暴露作者的信息，而很多作者为了自己的隐私安全，不愿意对外公布这些信息。通过本产品提供的“图片注册”功能，可以在一定程度上抑制上述问题。图片创作者需要在平台上进行图片的注册，平台将会返回一个注册图片，创作者进行发布注册图片，当普通用户在网上查找到图片时，若该图片是平台的注册图片，则用户可以通过平台查询到图片注册人，并通过本平台和作者取得联系。很明显，通过这样的方式不但可以辨识某人是否为图片作者，也可以避免作者隐私信息遭到泄露。

2.2.2 盗版追踪

数字图像的创作者除了可以嵌入鲁棒的版权水印以保护自身的权益外，还可以嵌入数字指纹以实现盗版追踪。当用户获得作者的认可时，作者将会拷贝图像给该用户，为了进行盗版追踪，作者会将受信用户和一个唯一标识的水印数据绑定在一起，这样的水印就被称为数字指纹。当作者在网上发现自己的数字图像作品被盗版传播时，即可获取作品中的数字指纹，以获知是哪位用户进行盗版传播，并绳之以法。

可以通过本平台提供的“不可见水印”功能，向图片中嵌入不可见的数字指纹，以达到盗版追踪的目的。将数字指纹通过本产品进行不可见水印嵌入，并且强烈建议作者输入密码，以避免数字指纹被他人任意获取。当作者在网络上发现自己的图像作品被盗后，作者可以在本平台中提取数字指纹，再找到对应的受信用户，便可获悉盗版源头。

平台尚未直接开通盗版追踪功能，需要作者自行记录数字指纹和其对应的受信用户，较为不便。在下一个版本中将会由平台提供全方位的基于不可见水印的盗版追踪服务，由服务器生成含数字指纹的图片，并由服务器分享给受信的微信用户。当作者发现网络上流传着自己的图像作品时，将该图像交给服务器的盗版追踪服务，将会为作者返回对应的受信用户。

2.3 目标用户

1).图像作者

图像作者在互联网平台上发布图片后，常常希望自己的图片可以得到广泛的传播， 又不希望自己的图片被滥用，损害自己的合法利益。图像作者可以通过平台提供的“可见水印”和“不可见水印”将自己的信息嵌入到图片中，对于不可见水印，普通用户可以通过平台中的“不可见水印”提取功能，获得图片作者留下的信息，以便和作者联系。当图片作者在“不可见水印”功能中，输入了密钥，意味着创作者不愿意别人使用自己的图像，但仍然加入水印，以保证自己的版权信息。

通过“不可见水印”和“可见水印”功能会留下作者的信息，在全网公开后，作者的信息容易被他人获得，造成作者被骚扰等麻烦。为了避免图像作者的隐私信息遭泄露，平台通过“图片注册”功能，图像作者可以在平台上注册的自己的创作图，普通用户可以通过图片作者发布的注册图片在本平台上向作者发送信息，图片作者视情况和其联系。

本平台为图像作者提供了第一层的水印上的版权保护，用户希望版权受到最大程度的保护，还需要自己保留原图，并在发表图片前进行著作权的登记，尽可能留下创作证据。平台

后期也会引入不局限于水印的，更全方位的版权保护的服务。

2).希望使用图片的用户

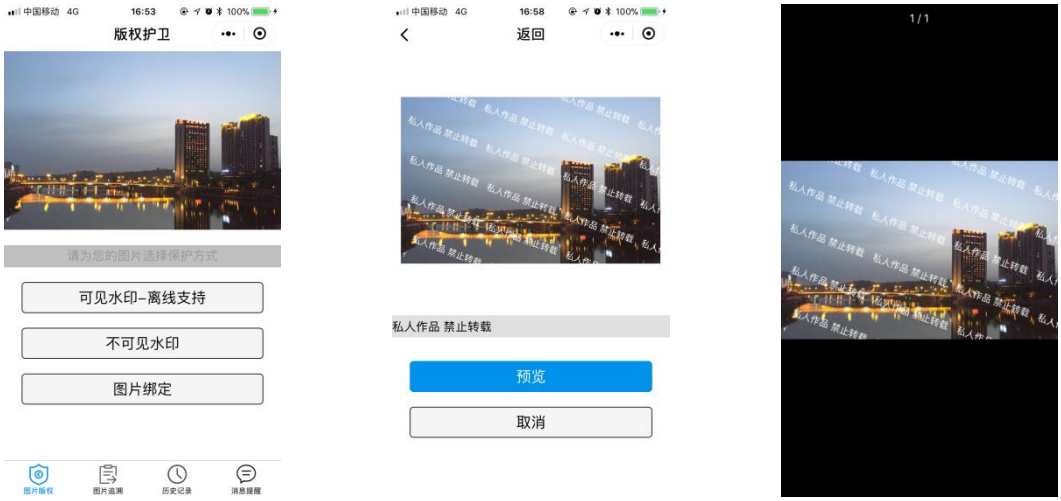
现今国内仍然处于普遍缺乏版权意识的时代，百度和谷歌等搜索引擎会爬到很多没有水印信息的图片，很多互联网用户在分享图片时也较为随意，若不加以注意，对来自互联网的图片的进行滥用，很容易让自己陷入版权纠纷。若是用户在使用前可以和图片作者取得联系，在引用图片前争取到作者的同意，则能极大的降低此类纠纷的发生。

“可见水印”或者“不可见水印”都是图片创作者将自己的联系方式等信息嵌入到图片中，用户看到作者的信息后可以主动与作者进行联系。为了避免图片创作者的信息随着图片的公开而遭泄露，通过“图片注册-图片追溯-消息发送”的功能，可以有效的避免图片创作者的信息遭到泄露，普通用户通过图片追溯的方式可以查询到作者，并向其发送消息，图片创作者视自身情况考虑是否和用户进行联系。

三、功能详解

3.1 可见水印

本产品虽然是以不可见水印技术核心的，但是为了更为平稳的向不可见水印技术进行过度，并且提供更广全方面的版权保护功能，因此同样也提供了可见水印的功能。可见水印的功能较为简单，并不会涉及到复杂的数学计算，因此在小程序端进行完成，并支持离线功能。在图 3-1 中给出了可见水印的相关页面。



(a).选择图片　(b).可见水印　(c).含可见水印图预览

图 3-1 可见水印

3.2 不可见水印

该功能可以往图像中嵌入不可肉眼察觉的水印，需要在选择了原始图片后，输入图片的标题(选填)，不可见水印文字内容(必填)，提取水印的密钥(必填)。由于不可见水印涉及到较为复杂的计算，通过手机来进行运算速度太慢因此当前版本会将图像发送到服务器进行不可见水印内容的嵌入，嵌入完成后会将含密水印返回。图 3-2 描述了嵌入的相关页面。

(a).选择原始图像

(b).输入必要信息

(c).成功图片预览

图 3-2 嵌入不可见水印

存在水印的嵌入操作，就会有水印的提取操作，需要在选择了图像后，输入水印密钥。图 3-3 描述了不可见水印提取的相关页面。

(a).输入密码

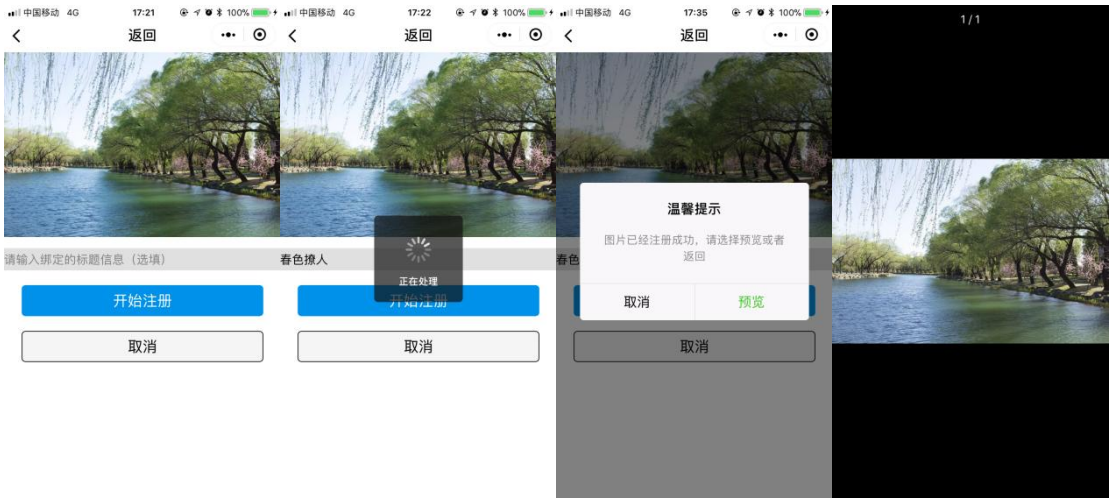
(b).提取水印中

(c).水印提取完成

图 3-3 提取不可见水印

3.3 图片注册和追溯

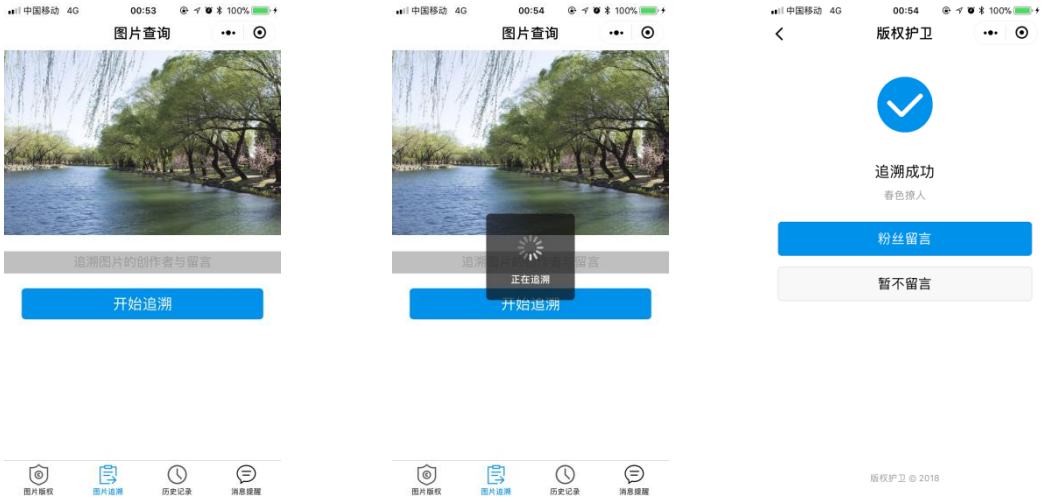
图片注册和追溯，是一个闭环的版权保护功能，当图片创作者在本平台上进行了照片的注册后，平台将会绑定图片和作者，并返回一个注册图片(该图片含水印)，作者可以将服务器返回的注册图片用于任何活动的使用。在图 3-4 给出了图片注册的相关页面。



(a).不可见水印　(b).输入标题并开始　(c).注册成功　(d).预览图片

图 3-4 图片注册

在平台上，用户对注册图片进行溯源，能够找到该图片所对应的注册者，并和该注册者取得联系。在图 3-5 中给出了图片追溯的相关页面。由于微信是一个较为私人敏感的平台，因此为了避免注册者的隐私遭到泄露，因此追溯成功是不会显示关于注册者任何相关信息的。



(a).选择追溯图片　(b).开始追溯　(c).追溯完成

图 3-5 图片追溯

3.4 留言收发

对于在平台上注册的图片，用户能够在平台上进行图片追溯，以找到图片的注册者，当找到注册者以后，用户可以向注册者发送消息。为了避免消息发送者的隐私泄露，本平台不会发送关于发送者的任何隐私，因此需要用户在消息中写入自己的愿意透露的别名和联系方式等信息。在留言发送成功后，注册者会接收到未读的留言，注册根据留言发送者中透露的信息，自行选择是否和该用户进行联系。在图 3-6 中显示了发送留言以及留言查阅的相关页面。当前版本还未引入黑名单功能，在后续会加入。

(a).编写并发送留言　　(b).查看留言

图 3-6 发送留言

3.5 历史记录

考虑到用户进行了水印嵌入操作以后，由于某些原因导致了含水印图像的丢失，因此在服务器上保存了含水印的图像，用户可以通过历史记录查看所有操作的含水印图像，需要注意的是服务器中没有保存用户提供的原图，历史记录中也不会提供原图的记录，并且也不会包含可见水印图像的记录。在图 3-7 中给出了历史记录的相关页面。

图 3-7 历史记录

四、技术开发方案

4.1 整体架构

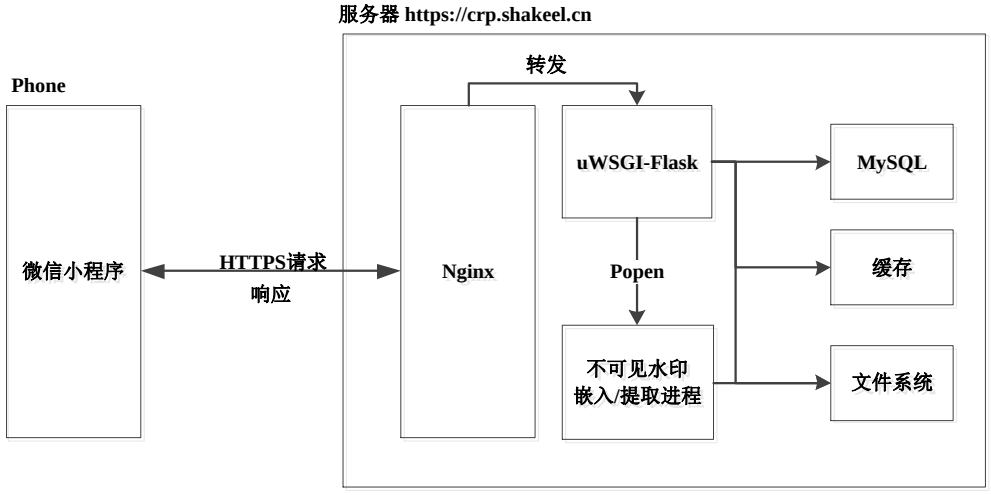


图 4-1 系统整体框图

4.1.1 服务器端

服务器后台将会提供 HTTPS 接口，以便小程序获得“不可见水印”和“图片注册”的相关服务。后台采用 Python-Flask 进行开发，通过 uWSGI 运行，为了更好的支持 SSL 功能，搭建了 Nginx 服务器接收 HTTPS 请求，并将请求转发给 uWSGI 以处理。

Python 后台在产品中的主要作用是接收和返回网络 IO 任务以及相关数据的出库入库，

而对于不可见水印的嵌入和提取过程，这是一个 CPU 密集的任务，考虑到 C++在 CPU 密集型任务上处理的优势，因此对于不可见水印的嵌入和提取请求，Python 将会转交给 C++进程进行处理。目前转交给 C++处理的形式较为简单和粗暴，是通过 Python 调用 C++进程的形式进行的，这样的形式缺点主要在于对于高并发情况下，大量的进程创建和销毁，这将会成为性能瓶颈，因此后期考虑通过 C++进程进行端口监听，Python 通过 Protobuf 将任务序列化，并交给 C++进程，C++进程接收到 Python 的任务后，提交任务给线程池来处理不可见水印提取和嵌入的任务，图 4-2 所示。改进的机制不但避免了大量的进程创建销毁所需要花费的代价，并且限制了进程的无限制创建，通过线程池机制可以在高并发环境下削峰填谷。

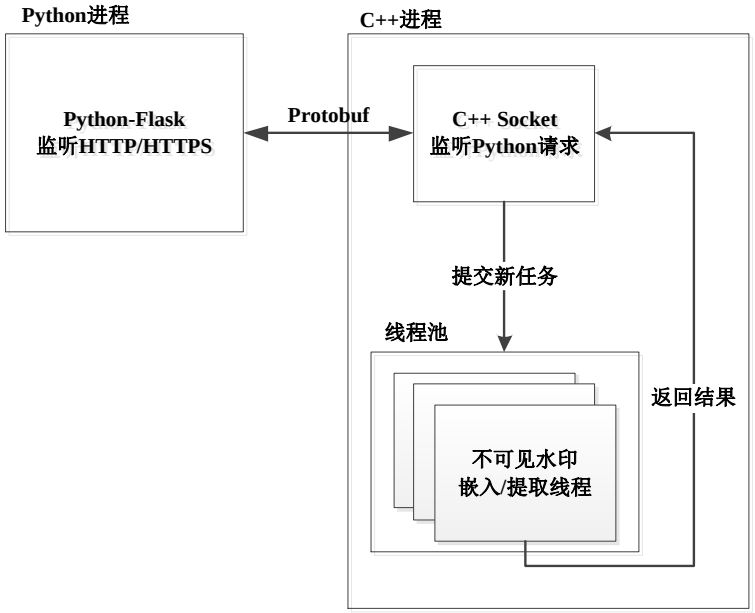


图 4-2 改进的 Python 转发 C++机制

4.1.2 小程序端

微信小程序端向服务器发送 HTTPS 请求，传送相关的数据，并根据服务器反馈的数据进行小程序端的页面渲染。每个页面包括有：页面逻辑 JS，页面结构 WXML，页面样式表 WXSS 和页面配置 JSON。其中，WXML 用来编写页面的标签和骨架，主要使用了一些小程序自己的组件，使用 WXSS 提供的样式和同时与 JS 的事件进行响应，为了方便数据渲染在历史记录和消息提醒中采用了列表的形式；WXSS 中放了一些样式容器，从而对布局进行控制，主要采用的是 FLEX 布局，本产品除了使用自己构建的样式以外，还使用了微信官方设计团队提供的 WeUI 基础样式库；该小程序的 JSON 比较简单，主要是返回键的设置；JS 主要是点击事件的相应方法，同时使用了官方提供的 API，实现对小程序页面的准确渲染。具体框架如图 4-3 所示。

图 4-3 小程序端框图

4.2 解决方案

4.2.1 小程序端

1）会话机制

微信小程序的会话包括：会话未过期并向服务器发送请求刷新会话，和会话已经过期并提示用户重新登录。小程序登录时系统会首先检测缓存中是否存在 sessionId（会话 ID），如果已经存在，那么合并会话可开始使用，如果在这个过程中抛出了 1002 异常，说明该会话已经过期，那么用户此时可以选择重新登录或者离线使用，如果选择了重新登录，为了避免不同系统中同时使用，首先会在缓存中检测设备的唯一 ID 是否存在，如果存在直接登录，反之则获取设备再进行登录。如果小程序登录时没有监测到 sessionId，说明会话不存在，那么这个时候用户可以选择离线使用或者同意登陆。具体的流程如图 4-4 所示。

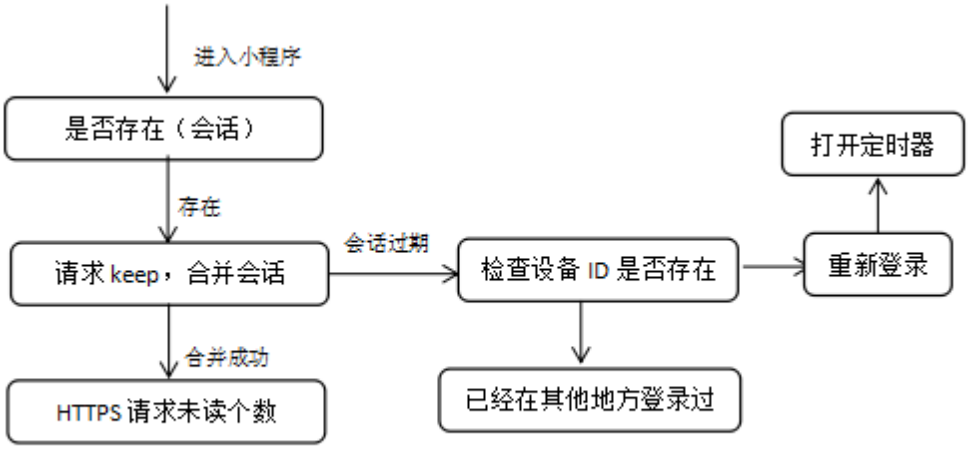


图 4-4 小程序会话处理流程图

2)缓存机制

该产品为了在离线的情况下也可以查看历史记录和消息提醒中的信息，小程序端设置了多个缓存，把服务器传送过来的数据以二维数组的形式放在缓存中，需要的时候再进行页面渲染。此外，小程序中使用了多处的系统数据缓存，功能相当于全局变量，但优于全局变量。

3)定时器机制

定时器的作用是监测用户使用该小程序的活跃程度，从而进行会话的刷新。小程序进入之后定时器设置一分钟，在一分钟里面，如果有出现点击事件，那么就进行会话合并，直到小程序结束。

4)消息提醒 HTPPS 请求

消息提醒采用了官方提供的接口 setTabBarBadge，需要时刻对邀请人发来的信息进行刷新，因此需要在每个页面都进行一次 HTTPS 的请求，获取最新的未读提醒个数，从而显示在界面上。由于该请求放在 onShow 里面，为了避免与会话发生冲突，所以，需要确定 sessionId 存在的前提下才发送获取未读提醒的请求，因此在 onShow 页面进来的时候，向发一次定时器，使会话合并，要是会话过期则可以选择重新登录。

4.2.2 服务器端

1). 会话机制

Session(会话)是 Web 架构中常见的机制，通过在 cookie 中保存 sessionId 的形式来实现，用来在服务器上保存当前会话的重要数据，例如登录信息和配置信息等。在微信小程序中，并没有实现 Cookie 机制，进而无法支持 Web 的 Session 机制，因此需要我们自己设计 Session

机制。当小程序每次登录的时候，都会发送一个会话建立请求到服务器，该请求中包含了微信登录凭证 code 和设备唯一码 did，便于服务器获取建立会话并获取该会话所对应的微信 ID。服务器在会话建立的时候，会立即将微信 ID 放到会话的缓存中，以便获取后续每个请求所对应微信 ID。

同一个设备上的同一个微信用户，若已经和服务器建立了会话后，由于某些原因(例如 sessionId 丢失)将会再次尝试和服务器建立会话，服务器将会进行会话合并，即返回该服务器和该微信小程序已有会话的 sessionId。微信自身虽然可以避免同一个微信用户在不同的设备上进行登录，但由于本产品是基于 HTTPS 的，因此极易被模拟登入，会有同一个微信用户在不同的设备上登录本平台的风险，进而导致同一个会话服务于多个设备(会话合并造成的)，因此需要设备唯一码 did 进行标识。微信小程序本身无法获取 did，因此需要到服务器上获取 did。服务器的 did 接口每次调用都会返回一个不一样的 ID 号，小程序在第一次登入的时候将会缓存该 ID 号，作为自己的设备唯一码，以后每次会话建立请求都应该带上该设备唯一码。图 4-5 是服务器接收到会话建立请求后的处理流程图。

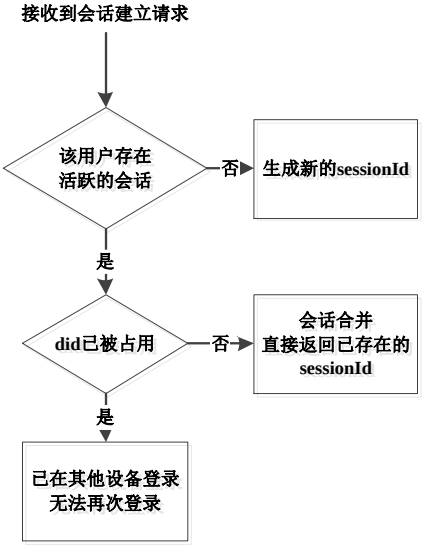


图 4-5 会话建立流程图

服务器有两种方式销毁会话，一种是服务器提供的会话销毁接口，在小程序退出的时候应该调用会话销毁接口。另一种是超时自动销毁，当会话建立超过 10 分钟，服务器会自动销毁会话。会话合并以及会话保持都能刷新会话持续时间。持续时间是为了避免小程序端由于某些原因，导致会话销毁请求无法发送出去，进而占用服务器资源。小程序端应该持续检测用户的活动情况，当用户存在活跃点击的时候，小程序将会周期性发送会话会话保持请求，进而方便刷新会话超时时间。图 4-6 是整个会话的生命周期时序图。

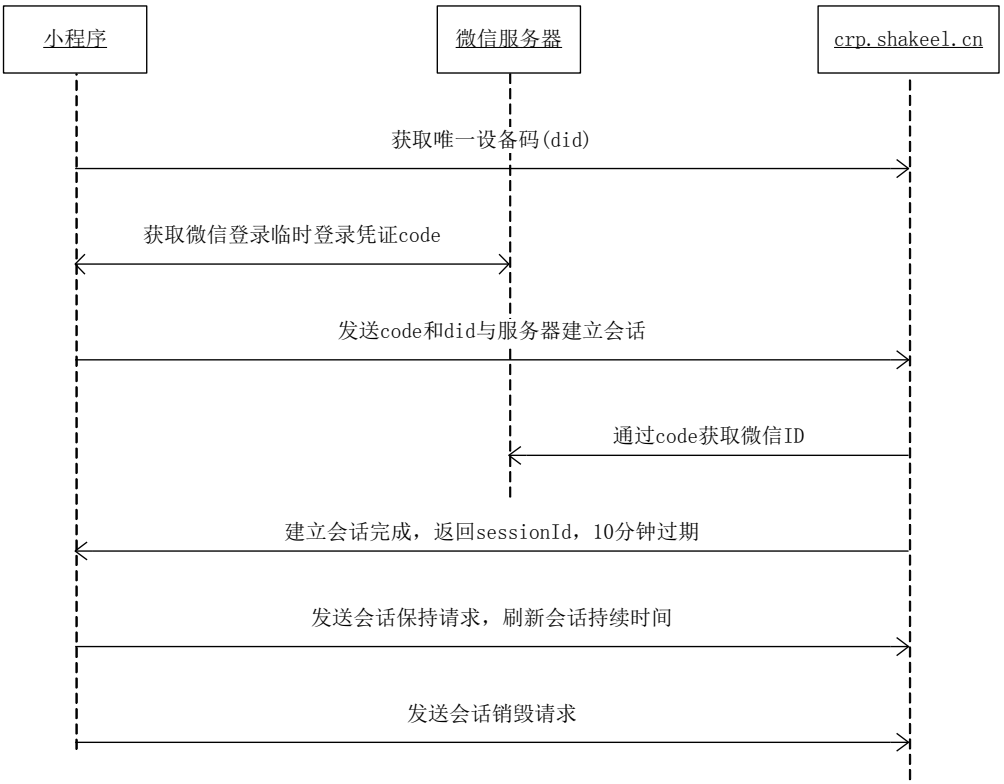


图 4-6 会话生命周期

2). 不可见水印嵌入机制

一张图片的大小有限，而嵌入的数据无限制增大时，将会严重影响图像质量以及增大图片文件的大小。“不可见水印”为了避免对图像质量影响较大或是增大图像二进制数据大小，因此采用了隐藏 *imgnum* 的方案，将 md5(*imgnum*)和不可见水印的内容绑定在一起。

imgnum 是一个 64 比特的数据，并且前 16 比特为校验位，用于标志该 *imgnum* 是否为平台所生成的不可见水印，而 md5(*imgnum*)是图像的 ID 号，即 *imgid*。在嵌入不可见水印时，为了避免图片已经在平台上进行过水印嵌入或是图片注册，将会首先提取 *imgnum*，首先检查校验位，校验不通过则认为该图片不含本平台的水印，可以进行嵌入。若校验通过，再检查数据库中是否存在该 *imgid*， 若存在则认为该图像已经在图片嵌入过数据，不能再次进行嵌入。不可见水印在提取时，在提取 *imgnum* 后，也需要先检查校验位，校验不同则在平台上没有因此数据，若校验通过则到数据库中去寻找 *imgid*，若没有找到该同样认为该图像没有因此数据，否则返回图像不可见水印。

3). 图片追溯的隐私保护机制

图片注册相比于不可见水印的最大的优点是在保护版权的同时，避免图片创作者在图片中加入自己的信息，导致的隐私泄露。图片追溯可以让意图使用图片的人在平台上和作者取得联系，作者视情况自行回复，避免了隐私的泄露。因此在整个图片注册，图片追溯和消息发送的流程中，都不应该泄露除了图片本身以外的任何有关作者的信息。

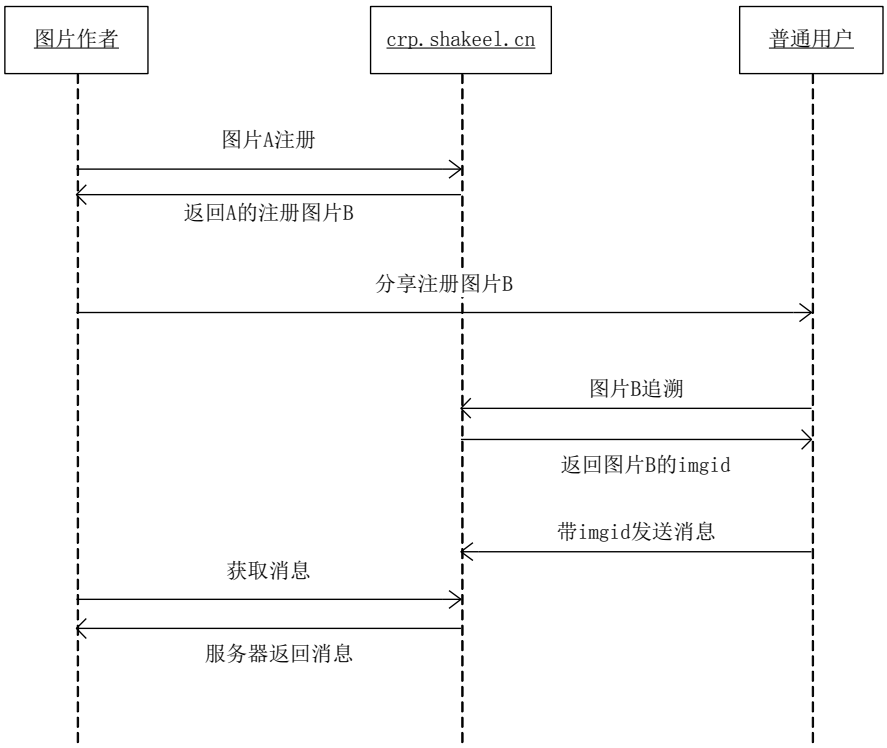


图 4-7 图片追溯时序图

在图 4-7 中给出整个流程时序图，首先图片注册就会先生成一个 *imgid*，将其作为水印信息嵌入到图片中返回给小程序，并将 *imgid* 和微信 ID 在数据库中绑定在一起。图片追溯时，后台将会提取出水印作为 *imgid*，并且当 *imgid* 在数据库中存在时，则认为图片经过了注册，能够追溯到原作者，这时将会返回小程序该图片的 *imgid*。用户在追溯成功后，可以发送消息，消息发送会带上追溯成功时返回的 *imgid*，服务器接收到了消息发送的请求，会将 *imgid* 所对应的微信用户查询出来，再将消息发送给该微信用户。很明显，整个追溯过程都不会泄漏关于图片注册的任何信息，都是由希望使用图片的用户发起的所有操作，并等待图片作者的回复，这样确保了图片作者的隐私不被泄露。

4). 变分辨率问题

平台在进行图片注册或是嵌入不可见水印时，会在保持宽高比的条件下，对分辨率进行变更。一次是在小程序发送图片注册或嵌入不可见水印请求时，对上传上来的图片进行分辨率变更， 另一次是算法进程生成了含水印图像时，会额外生成一张小分辨率的图片。

在图片注册或嵌入不可见水印时，会将图片变分辨率主要是基于两点考虑：a).水印鲁棒性问题。实际嵌入的数据为了达到一定的鲁棒性会进行扩频，图像分辨率越大，将会有更大的扩频空间，以增加鲁棒性。b).微信转发图片对图像分辨率的变更。经过反复的测试发现，微信在发送图片时，会以一些规律对图片在保持宽高比的情况下进行分辨率的变更。含水印图片在变更分辨率后鲁棒性会变差，为了避免这样的情况出现，平台将会首先对图片分辨率进行变更，避免微信转发图片时对图片分辨率的变更导致提取水印时的困难。

当小程序发送获取消息和获取历史记录的请求时，为了让用户知道历史记录对应的图片以及消息提醒所对应的图片，服务器将会返回对应图片的 URL。因为含水印的图片都是非常大的，不但影响传输和渲染效率，也增大小程序的缓存时的容量压力。由于移动端设备童通常较小，历史记录和消息提醒的图片预览并不需要获得原图，因此在图片注册和嵌入不可见水印以后，都会生成小分辨率图像，方便历史记录和消息提醒中的预览。后期考虑为了适配更多种类的移动端设备，例如平板，大型号手机等，将会引多分辨率的方案，并由小程序端指定需要哪种分辨率的图片，以达到自适应设备的功能。

4.3 不可见水印

为了更直观的表示不可见水印的嵌入和提取，在图 4-8 中描述了算法系统框图。对于不可见水印的嵌入，发送端向算法提供所需要的原始图像、待嵌入水印和密钥后，将会生成含水印图像。含水印图像经过了可能含攻击的信道，在接收端提供密钥后，通过提取算法提取。鲁棒的不可见水印要求图片在一定程度攻击的情况下，仍然可以较好的恢复出水印数据。

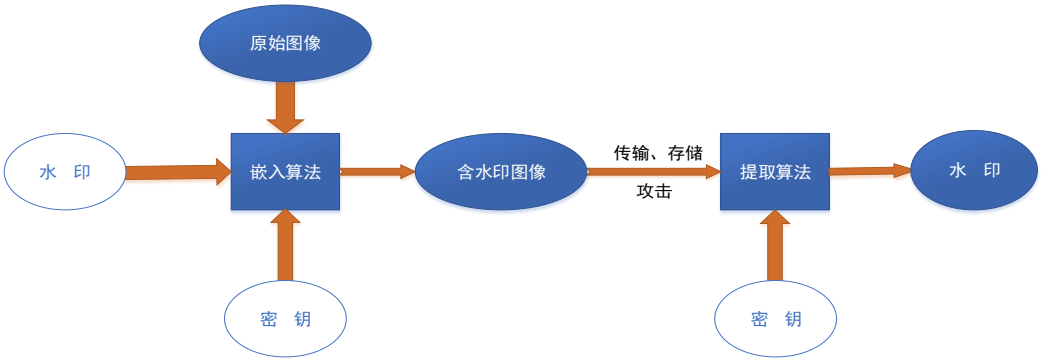


图 4-8 嵌入提取框架

4.3.1 不可见性

数字水印技术按表现形式为可见水印和不可见水印，不可见水印是无法用肉眼看见，可见水印前者如图像上的 LOGO 是肉眼可以看见的水印，即可见水印。

数字水印的不可见性就是数字水印嵌入到载体图像中后通过人的感知系统是察觉不到的，对于图像数字水印来说就是嵌入水印的图像和原始图像看起来是一样的，即看不到数字水印的存在；嵌入水印后不会引起原来数字作品明显的图像质量下降，而且应不影响载体图像的正常使用；不可见水印之所以肉眼不可见是因为人眼的视觉冗余，我们看到的数字图像是由许多的像素点组成的，我们可以轻微的他修改每个像素的值来实现水印的嵌入，而且嵌入水印的图像和原始的图像肉眼看起来并没有差别，不影响图像的使用。水印的提取就是更根据算法设计的规则，从图像像素中提取信息。

4.3.2 水印鲁棒性

数字水印实现版权保护的原理在于，将能证明产品来源的信息用相应的算法嵌入到图像中，当发生版权纠纷时，可以通过算法从图像中提取信息，以此来证明图像的来源，达到本版权保护的目的。在图像传输和存储过程中，不可避免的会对图像进行处理，而且存在对图像进行恶意篡改的可能，如果在经过处理后不能从图像中提取有效地水印信息，那就无法证明图像的来源。因此，鲁棒数字水印要求经过无意或恶意的图像处理过程后，数字水印仍能保持部分完整性并能被准确鉴别。数字水印的鲁棒性指的是，含水印图像在受到攻击后对水印信息的恢复情况，通常以误比特率来进行判断，认为含密水印提取出来的数据误比特率越低，该算法的鲁棒性越强。常见的攻击包括图像压缩、旋转、剪切、缩放(改变分辨率)、涂抹等，这些操作都会对改变图像的像素数据。需要考虑的攻击类型：

1).重编码问题

在图像的传输和存储中，为了提高传输效率和节约存储空间，都会先对图像进行压缩编码，像素数据编码成比特流，再按字节处理就可以方便快捷的传输和存储；在微信传输的 JPG 的图像就是经过 JPEG 压缩编码后的图像，JPEG 编码属于有损压缩，即经过 JPEG 编码，再解码后得到的像素数据与原始未经过 JPEG 编码的像素数据是不一样的，即在编码、解码的过程中存在像素的数据的丢失。而水印的嵌入和提取都是在图像像素上进行操作，所以必

须考虑编解码对水印提取带来的影响。

2).涂抹问题

涂抹即在图像用其他颜色对部分像素进行遮盖，如图 4-9。当用其他颜色对图像进行涂抹后，涂抹区域的像素值就会被更改，如果被涂抹的区域嵌有水印，那该部分的水印就会丢失，从而影响水印的提取效果。

(a).Lena 涂抹

(b).Vegetables 涂抹

图 4-9 涂抹攻击示意图

对于可见水印，很容易通过图像修复技术等方法将水印部分进行修复，这也是一种涂抹，导致水印的丢失。在不可见水印技术中，水印会通过位置置乱以及扩频增加冗余的技术来保障图像具备一定的抗涂抹能力，提升图像版权的安全性。

3).缩放问题

在微信上以非原图的形式传输图像时，微信端会在某些情况下对图像进行缩放，缩放的规则是保持宽高比不变，将宽、高中较小的那个放大或缩小到 1080，另一边同比例缩放。因为水印嵌入和提取都是在像素点上进行的，在全部像素中根据密钥选择像素值进行嵌入和提取，图像缩放后，像素的总数发生改变，同一个密钥可能选择不同的像素点，这样提取出来的水印就不是之前嵌入的水印。

4.3.3 核心技术

1).小波变换

为实现嵌入不可见水印，在尽量减小失真的情况下增强水印的鲁棒性，保证水印的提取，以达到图像溯源的效果；采用多级离散小波变换(DWT)，在低频分量上进行水印嵌入。小波变换对不同的频率在时域上的取样步长是可调节的，即在低频时小波变换的时间分辨率较低，而频率分辨率较高；在高频时小波变换的时间分辨率较高，而频率分辨率较低，这正符合低频信号变化缓慢而高频信号变化迅速的特点。

DWT 因其符合人眼视觉系统(HVS)特性，从而在数字水印中得到广泛应用。单通道的灰度图像经过 1 级 DWT 后可以分解为 4 个分量，分别是低频分量(cA)、水平中频子带(cH)、垂直中频子带(cV)和高频子带(cD)，多级分解在上级低频子带的基础上再次分解。其中低频分量 cA 代表了原始图像的大部信息，是最接近原始图像的分量，考虑到图像传输、存储时，微信或其他平台会对图像作压缩、调整分辨率等处理，这些处理会影响的水印的提取；因此为了能完整无误的提取水印，选择在低频分量上嵌入水印。

2).扩频水印

因为本设计嵌入的水印所占的字节数较小，如果直接嵌入水印数据，还有剩余的子块没有嵌入数据，为了能利用载体图像的全部系数，并且进一步增强水印的鲁棒性，采用扩频技术对水印预处理。

扩频技术的特点是传输信息所用的带宽远大于信息本身带宽时，扩频通信技术在发射端

以扩频编码进行扩频调制，在收端以相关解调技术收信息，这种技术的目的和作用是在传输信息之前，先对所传信号进行频谱的扩宽处理，以便利用宽频谱获得较强的抗干扰能力、较高的传输速率。

在嵌入前，先将水印的每个比特位重复 P 倍，如原始水印为 0110；二倍扩频后得到 00111100，然后再将扩频后的水印嵌入到载体图像中。提取水印时，将提取得到的比特位按扩频的倍数进行解调，如上例，每两个比特位为一组，每组中哪个值频率大，该组就以哪个数为最终提取的水印；这样做的优点在于，对含水印图像进行攻击后，如果只有少数的值发送改变，依然可以恢复水印，要使提取的水印发送错误，就要让超过一半的值发生改变，就必须用更强的攻击才行。

4.3.4 实验

为了快速验算法的效果(失真和提取)，先在 MATLAB 下仿真观察实验效果，再在 Windows 下借助 OpenCV 跨平台图像处理库实现 C++程序的编写，最后将编译好的代码移植到 Linux 平台下编译运行，完成算法的设计和实现。

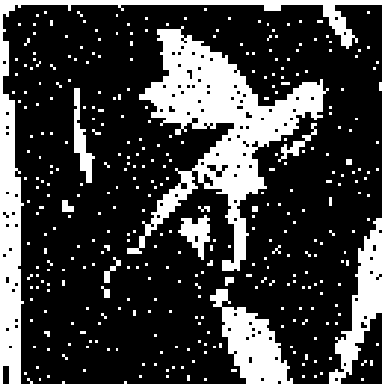
将一个图像进行二值化，再将其嵌入并进行提取，这能只管的反应水印的不可见性以及鲁棒性。在 Matlab 中仿真结果如图 4-10，图(a)是原始图像，图(b)是待嵌入的水印图像，图(c)是嵌入水印后的图像，并且是经过重编码的含水印图像；从直观上看嵌入水印前后图像没有差别，所以不会影响图像的使用；图(d)是提取出来的水印图像，因为存在重编码的问题，所以不能完全恢复原始水印。图(e)是未经重编码的含水印图像，图(f)是从未经重编码的含水印图像中提取的水印。从实验结果中可以看出，直观上分辨不出提取的水印与原始的水印图像有什么区别。

(a)原始图像

(b)原始二值水印

(c)原始图像

(d)原始二值水印

(e)JPEG 压缩的含水印图像　　(f)提取水印

图 4-10 误比特率试验采用的图像

为了更好地观察含水印图像受到 JPEG 重编码攻击后，不可见水印的提取的鲁棒性，在表 5-1 中给出了更详细的实验数据；图 A、图 B 和图 C 为三幅测试序列，嵌入的水印为原始的二值水印图像。未扩频误比特率表示没有对原始水印做扩频处理，提取水印的误比特率；扩频误比特率表示对水印进行扩频预处理后的，再嵌入水印的误比特率；压缩未扩频误比特率表示，没有对水印进行扩频处理，对含水印的载体图像作压缩处理后，从中提取水印的误比特率；压缩扩频误比特率表示对对水印做了扩频处理，并对含水印的载体图像作压缩处理后，从中提取水印的误比特率。

(A).Lena　　(B)house　　(C)vegetables

图 4-11 误比特率试验采用的图像

表 4-1 误比特率实验表

载体图像编号	未扩频误比特率	扩频误比特率	压缩未扩频误比特率	压缩扩频误比特率
A	2.771%	0.085%	5.444%	0.476%
B	3.479%	0.164%	7.941%	0.769%
C	5.194%	0.726%	9.979%	2.533%

由上述实验结果可以证明基于二维离散小波变换，将小波变换后的系数分块，根据相邻块的相对大小实现不可见水印的嵌入，可以抵抗 JPEG 压缩、涂抹、缩放等攻击，采用扩频技术可以有效减低水印误比特率，提高算法的鲁棒性。为不可见水印的实现提供了理论支持，最终达到版权保护的目的。

五、团队的组成与分工

1).陈智隆(队长)

负责鲁棒水印核心算法编写，包括了：1).测试微信发送图片对图片本身的 JPEG 压缩和

变分辨率情况。2).查阅相关文献，进行鲁棒水印的 Matlab 仿真，并测试在微信发送后水印的提取情况。3).采用 OpenCV 库，将鲁棒水印算法从 Matlab 移植到 C++平台，并分别编译了 Windows 和 Linux 版本的动态链接库，以便 Python 调用。

2).李素静

负责小程序端的界面显示以及所有交互效果，包括了：1).了解并熟悉小程序端开发组件和工具。2).设计并开发了小程序的显示界面。3).完成人机交互效果，优化用户体验。

3).卢帅吉

负责系统架构的设计以及系统的开发与部署，包括了：1).通过腾讯云架设服务器运行环境，如域名备案，Nginx 搭建，Python-Flask 环境搭建，SSL 注册等。2).设计服务器的 HTTPS 接口。3).开发服务器 CRUD 业务代码。