从图像的整个过程而言，采集-》传输-》处理—》显示（至少要和电视机播放标准的帧数一样——每秒25帧），在图像处理的时候会遇到这几个方向的知识：

**大数据优化网络**，通过对网络以及信号范围内的基站进行分析，寻找最佳的传输点以及制定传输路线。

**弱信号提取**，图像采集到的有效信号经过噪声，整个波段就会变得窄而高，于是需要使用弱信号处理算法将有效的部分利用起来，如果直接显示，那么得到的图像将会有着雪花甚至左右摇摆。

**多视点处理**，将几个非广角摄像头通过图像处理算法得到一个广角拼接图像。运行以后其实得到的就是广角视频。多视点视频 (multi-view video)是一个近几年迅速崛起并快速发展的研究新领域。多视点视频通过在场景中放置多台摄像机，获取整个场景的多个视点的视频信号，并将这些视频信号提供给用户，从而可以实现三维场景、自由视点切换以及场景漫游等功能。

**多源图像融合**，图像融合（Image Fusion）是指将多源信道所采集到的关于同一目标的图像数据经过图像处理和计算机技术等，最大限度的提取各自信道中的有利信息，最后综合成高质量的图像。这个老师举得例子是，通过两个特定波段内的光波收集器，收集到光波以后，经过处理得到夜间的图像。出人意料的效果。

**我觉得值的一读的论文文章：A general framework for image fusion based on multi-scale transform and sparse representation**

**计算机视觉**

**伪彩色处理**，图像融合 中举得例子，得到的图像是灰度图，若有必要通过判断算法进行上色的话就是伪彩色处理。

**张量噪声处理**，一种处理噪声的算法，但是对性能要求还是比较高的。一般的树莓派不能满足性能要求。

**压缩感知**，根据[麦奎斯特采样定理](http://www.baidu.com/link?url=Nr3Z3Wi5UBtFaz5XYNfsPkdkhAFl_4llJiLsIEv08V6ufvDzxKPGTUR7kV7gytH--3jum02jzs0-PHF2lDC5SefVnSb_SEo-AIYwnjEPaQfKchPZoDaij9MWgZRqznPr)，在进行模拟/数字信号的转换过程中，当采样频率fs.max大于信号中最高频率fmax的2倍时(fs.max>2fmax)，采样之后的数字信号完整地保留了原始信号中的信息，一般实际应用中保证采样频率为信号最高频率的5～10倍。压缩感知的话就尝试使用更小倍数的频率尽可能得到数据。

**协同计算**，因为“张量噪声处理”以及“压缩感知”需要大量的计算性能，所以可以使用几十个树莓派进行组合，得到一个高性能的协同计算平台。传闻30个raspberry B+进行组合以后的性能，能够有中等计算机的水准。

**视觉显著性检测**，指通过智能算法模拟人的视觉特点,提取图像中的显著区域(即人类感兴趣的区域)。

<http://www.ituring.com.cn/tupubarticle/2024>（Python计算机视觉编程）

1080P是高清，1280P\*720P 以下就是标清。

信道的传输容易受到大风、大浪这样的干扰。所以海洋上的远距离信道传输除了要解决以为地球曲率带来的绕传的信号影响，还需要解决因为恶劣的海洋环境对信道的影响。

图像灰度变换中一个非常有用的例子就是直方图均衡化。直方图均衡化是指将一幅图像的灰度直方图变平，使变换后的图像中每个灰度值的分布概率都相同。在对图像做进一步处理之前，直方图均衡化通常是对图像灰度值进行归一化的一个非常好的方法，并且可以增强图像的对比度。

在这种情况下，直方图均衡化的变换函数是图像中像素值的累积分布函数（cumulative distribution function，简写为 cdf，将像素值的范围映射到目标范围的归一化操作）。

from PIL import Image

from numpy import \*

def histeq(im,nbr\_bins=256):

""" 对一幅灰度图像进行直方图均衡化"""

# 计算图像的直方图

imhist,bins = histogram(im.flatten(),nbr\_bins,normed=True)

cdf = imhist.cumsum() # 累积分布函数

cdf = 255 \* cdf / cdf[-1] # 归一化

# 使用累积分布函数的线性插值，计算新的像素值

im2 = interp(im.flatten(),bins[:-1],cdf)

return im2.reshape(im.shape), cdf 该函数有两个输入参数，一个是灰度图像，一个是直方图中使用小区间的数目。函数返回直方图均衡化后的图像，以及用来做像素值映射的累积分布函数。注意，函数中使用到累积分布函数的最后一个元素（下标为 -1），目的是将其归一化到 0...1 范围。你可以像下面这样使用该函数：

图像的主成分分析（PCA）

PCA（Principal Component Analysis，主成分分析）是一个非常有用的降维技巧。它可以在使用尽可能少维数的前提下，尽量多地保持训练数据的信息，在此意义上是一个最佳技巧。即使是一幅 100×100 像素的小灰度图像，也有 10 000 维，可以看成 10 000 维空间中的一个点。一兆像素的图像具有百万维。由于图像具有很高的维数，在许多计算机视觉应用中，我们经常使用降维操作。PCA 产生的投影矩阵可以被视为将原始坐标变换到现有的坐标系，坐标系中的各个坐标按照重要性递减排列。

为了对图像数据进行 PCA 变换，图像需要转换成一维向量表示。我们可以使用 NumPy 类库中的 flatten() 方法进行变换。将变平的图像堆积起来，我们可以得到一个矩阵，矩阵的一行表示一幅图像。在计算主方向之前，所有的行图像按照平均图像进行了中心化。我们通常使用 SVD（Singular Value Decomposition，奇异值分解）方法来计算主成分；但当矩阵的维数很大时，SVD 的计算非常慢，所以此时通常不使用 SVD 分解

from PIL import Image

from numpy import \*

def pca(X):

""" 主成分分析：

输入：矩阵X ，其中该矩阵中存储训练数据，每一行为一条训练数据

返回：投影矩阵（按照维度的重要性排序）、方差和均值"""

# 获取维数

num\_data,dim = X.shape

# 数据中心化

mean\_X = X.mean(axis=0)

X = X - mean\_X

if dim>num\_data:

# PCA- 使用紧致技巧

M = dot(X,X.T) # 协方差矩阵

e,EV = linalg.eigh(M) # 特征值和特征向量

tmp = dot(X.T,EV).T # 这就是紧致技巧

V = tmp[::-1] # 由于最后的特征向量是我们所需要的，所以需要将其逆转

S = sqrt(e)[::-1] # 由于特征值是按照递增顺序排列的，所以需要将其逆转

for i in range(V.shape[1]):

V[:,i] /= S

else:

# PCA- 使用SVD 方法

U,S,V = linalg.svd(X)

V = V[:num\_data] # 仅仅返回前nun\_data 维的数据才合理

# 返回投影矩阵、方差和均值

return V,S,mean\_X

该函数首先通过减去每一维的均值将数据中心化，然后计算协方差矩阵对应最大特征值的特征向量，此时可以使用简明的技巧或者 SVD 分解。如果数据个数小于向量的维数，我们不用 SVD 分解，而是计算维数更小的协方差矩阵 XXT 的特征向量。通过仅计算对应前 k（k 是降维后的维数）最大特征值的特征向量，可以使上面的 PCA 操作更快。

图像的**高斯模糊**是非常经典的图像卷积例子。本质上，图像模糊就是将（灰度）图像 **I** 和一个高斯核进行卷积操作：

**I**σ = **I**\*Gσ

其中 \* 表示卷积操作；Gσ 是标准差为 σ 的二维高斯核。高斯模糊通常是其他图像处理操作的一部分，比如图像插值操作、兴趣点计算以及很多其他应用。

SciPy 有用来做滤波操作的 scipy.ndimage.filters 模块。该模块使用快速一维分离的方式来计算卷积。你可以像下面这样来使用它：

from PIL import Image

from numpy import \*

from scipy.ndimage import filters

im = array(Image.open('empire.jpg').convert('L'))

im2 = filters.gaussian\_filter(im,5)

上面 guassian\_filter() 函数的最后一个参数表示标准差。

图 1-9 显示了随着 *σ* 的增加，一幅图像被模糊的程度。*σ* 越大，处理后的图像细节丢失越多。如果打算模糊一幅彩色图像，只需简单地对每一个颜色通道进行高斯模糊：

im = array(Image.open('empire.jpg'))

im2 = zeros(im.shape)

for i in range(3):

im2[:,:,i] = filters.gaussian\_filter(im[:,:,i],5)

im2 = uint8(im2)

在上面的脚本中，最后并不总是需要将图像转换成 uint8 格式，这里只是将像素值用八位来表示。我们也可以使用：

im2 = array(im2,'uint8')