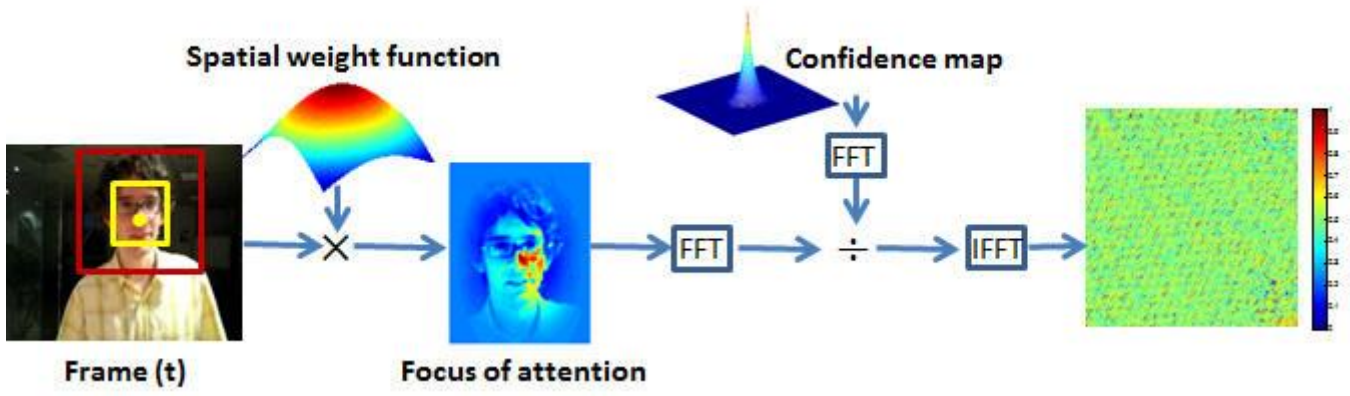
Correlation Filter in Visual Tracking 系列二：Fast Visual Tracking via Dense Spatio-Temporal Context Learning 论文笔记

原文再续，书接一上回。话说上一次我们讲到了 Correlation Filter 类 tracker 的老祖宗 MOSSE，那么接下来就让我们看看如何对其进一步地优化改良。这次要谈的论文是我们国内 Zhang Kaihua 团队在 ECCV 2014 上发表的 STC tracker：Fast Visual Tracking via Dense Spatio-Temporal Context Learning。相信做跟踪的人对他们团队应该是比较熟悉的了，如 Compressive Tracking 就是他们的杰作之一。今天要讲的这篇论文的 Matlab 源代码已经放出了，链接如下：

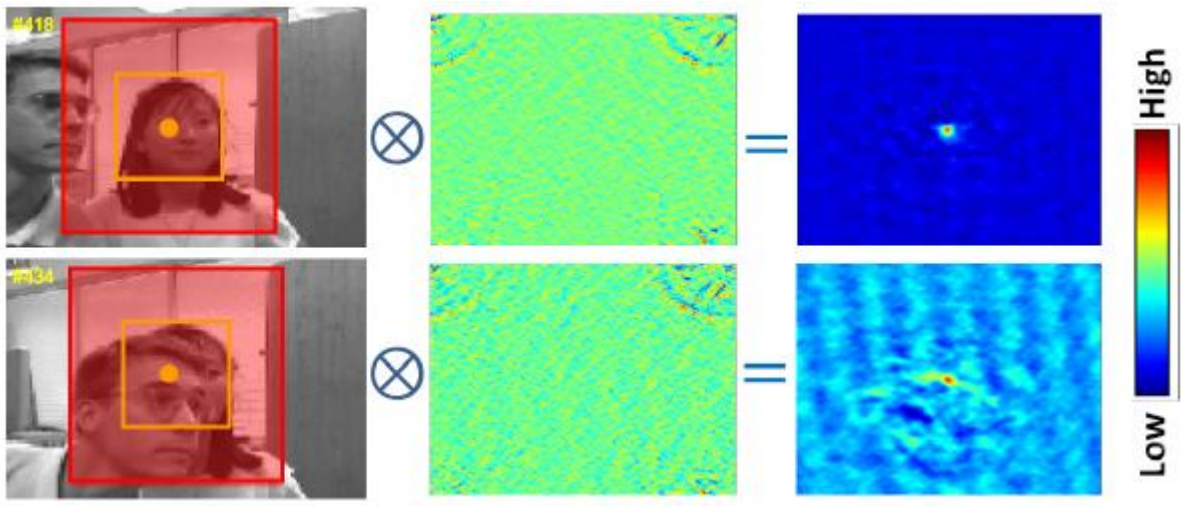
http://www4.comp.polyu.edu.hk/~cslzhang/STC/STC.htm

首先来看看他们的跟踪算法示意图：



看到更新方式，快速傅里叶变换什么的是不是很眼熟？没错，这篇论文其实与 MOSSE 方法基本是一致的，那么其创新点在哪了？笔者觉得，其创新点在于点，一是以密集时空环境上下文 Dense Spatio-Temporal Context 作为卖点；二是以概率论的方式包装了 CF 类方法;三是在模板更新的时候把尺度变换也考虑了进去。

那么什么是密集的时空上下文呢？其最朴素的思想可以用下面这个图来表达：在跟踪的过程中，由于目标外观变换以及遮挡等原因的影响，仅仅跟踪目标本身的话比较困难，但如果把目标周围区域也考虑进去（空间上下文），那么能够在一定程度降低跟踪失败的风险。以图中的例子来说，就是假如仅仅考虑目标本身（黄色框），那么在发生遮挡的时候，就难以实现跟踪，但是如果把周围的像素也考虑进去（红色框），那么就可以借助周围环境来确定目标所在。这是一帧的情况，假如考虑多帧情况的话，就对应产生了时空上下文。那么 dense 的说法从何而来？这一点我们后面再解释。



主要思想已经有了，下面我们来看如何用概率论进行理论支持。假设 $x \in R^2$ 为某一位置，o 为需要跟踪的目标，首先定义如下的 confident map 用来衡量目标在 x 出现的可能性：

$$m(\mathbf{x}) = P(\mathbf{x} \mid o) \tag{1}$$

然后定义 $X^c=\{c(z)=(I(z),z)|z \in \Omega_c(x^*)\}$ 为上下文特征集合，其中 x^* 代表目标位置，$\Omega_c(x^*)$ 表示在 x^* 点处两倍于跟踪目标大小的邻域，$I(z)$ 为 z 点的图像灰度值。这一公式的意思其实就是把 x^* 作为中心点，取其周围两倍于目标框大小的图像作为特征，如上图的红色框。然后我们利用全概率公式，以上下文特征为中间量把(1)展开：

$$\begin{aligned} m(\mathbf{x}) &= P(\mathbf{x} \mid o) \\ &= \sum_{c(\mathbf{z}) \in X^c} P(\mathbf{x} \mid c(\mathbf{z}), o) P(c(\mathbf{z}) \mid o) \end{aligned} \tag{2}$$

式(2)分为两项，左项 $P(x|c(z),o)$ 代表给定目标和其上下文特征，目标出现在 x 点的概率，右项 $P(c(z)|o)$ 则是某一上下文特征属于目标的概率，也就是目标的上下文概率先验了。右项的作用在于选择与目标外观相似的上下文，左项的作用在于在选择外观相似的同时也考虑出现在某一位置是否合理，避免跟踪过程中的漂移现象。

然后，因为在第一帧的时候，目标的位置是已知的，那么这时候就可以构造一个 confident map，使其满足距离目标越近可能性越高的性质。作者定义 confident map 的具体值为如公式(3)所示：

$$m(\mathbf{x}) = be^{-\left|\frac{\mathbf{x}-\mathbf{x}^\star}{\alpha}\right|^\beta} \tag{3}$$

其中 b,α,βb,α,β 都是经验常数。回想下上一篇我们讲的 MOSSE 方法，其实 m(x)m(x)就是我们讲的响应输出，只不过 MOSSE 直接用一个高斯形状，而这里用的是如(3)式的定义。另外，之前谈到本篇论文标题中有一“dense”字样，体现在哪呢？就体现在这个地方，对于目标附近每一个点，都可以用(3)式对其概率值进行定义。传统的跟踪方法可能是随机采样或者隔段采样，而这里因为每一个点都进行了概率值的定义所以就是 dense 了。但其实目前所有的 CF 类方法都是 dense sampling，而且这一个概念的明确提出应该是出现在后面会讲的 CSK 方法之中，只不过本篇作者将其改头换面成 dense spatio temporal learning 了。OK，闲话少说，接下来我们继续求解 P(x|c(z),o)P(x|c(z),o)和 P(c(z)|o)P(c(z)|o)。

先看 P(c(z)|o)P(c(z)|o)，是目标的上下文先验，定义为如下所示：

$$P(\mathrm{c}(\mathbf{z}) \mid o) = I(\mathbf{z})\omega_\sigma(\mathbf{z}-\mathbf{x}^\star) = I(\mathbf{z})\alpha e^{-\left|\frac{\mathbf{z}-\mathbf{x}^\star}{\sigma^2}\right|^2} \tag{4}$$

其就是目标框附近的图像灰度值的高斯加权和（换成其它特征也可以，后面另有一篇论文会谈到）。然后 P(c(z)|o)P(c(z)|o)有了，m(x)m(x)有了，就可以带入(2)求解 P(x|c(z),o)P(x|c(z),o)了，套路还是跟 MOSSE 一样，首先将 m(x)m(x)表示为 P(x|c(z),o)P(x|c(z),o)和 P(c(z)|o)P(c(z)|o)的卷积(互相关)，通过 FFT 转到频率域变为点乘运算，运算完后逆变换回空间域，找响应最大值的地方作为目标位置。 具体就是，设 P(x|c(z),o)=hsc(x−z)P(x|c(z),o)=hsc(x−z)，得

$$\begin{aligned} m(\mathbf{x}) &= be^{-\left|\frac{\mathbf{x}-\mathbf{x}^\star}{\alpha}\right|^\beta} \\ &= \sum_{\mathrm{c}(\mathbf{z})\in X^c} P(\mathbf{x} \mid \mathrm{c}(\mathbf{z}), o)P(\mathrm{c}(\mathbf{z}) \mid o) \\ &= \sum_{\mathbf{z}\in\Omega_c(\mathbf{x}^\star)} h^{sc}(\mathbf{x}-\mathbf{z})I(\mathbf{z})\omega_\sigma(\mathbf{z}-\mathbf{x}^\star) \end{aligned} \tag{5}$$

文中作者还强调了 hsc(x−z)hsc(x−z)是目标的位置与其环境上下文之间相对距离和方向的衡量，并且不是对称函数。

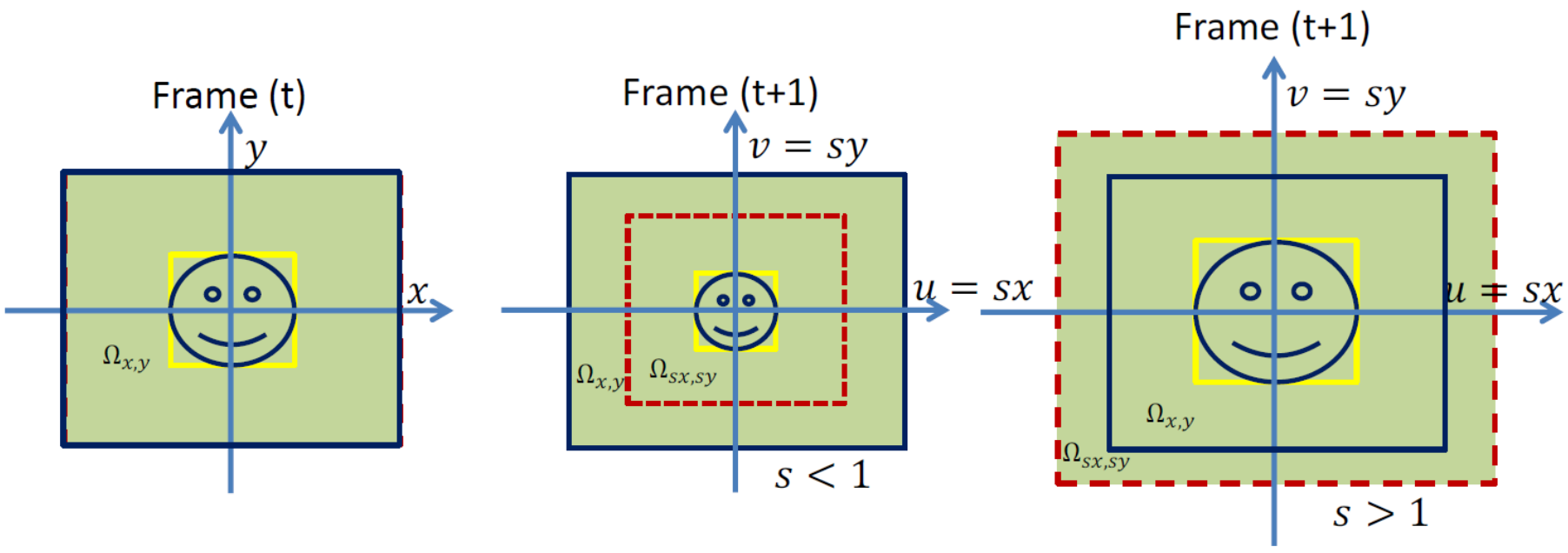
另外，根据卷积 f⊗gf⊗g 的定义：

$$\begin{aligned} (f\otimes g)(t) &= \int f(\tau)g(t-\tau)d\tau = (g\otimes f)(t) = \int f(t-\tau)g(\tau)d\tau \\ (f\otimes g)(m) &= \sum_n f[n]g[m-n] \end{aligned} \tag{6}$$

所以(5)式其实就是一卷积（xx 就是 tt 或 mm，zz 就是 ττ 或 nn），根据卷积定理：

$$\begin{aligned} &\mathcal{F}(m(\mathbf{x})) = \mathcal{F}(h^{sc}(\mathbf{x}))\odot\mathcal{F}(I(\mathbf{x})\omega_\sigma(\mathbf{x}-\mathbf{x}^\star)) \\ &\Rightarrow h^{sc}(\mathbf{x}) = \mathcal{F}^{-1}\frac{\mathcal{F}(m(\mathbf{x}))}{\mathcal{F}(I(\mathbf{x})\omega_\sigma(\mathbf{x}-\mathbf{x}^\star))} \end{aligned} \tag{7}$$

与 MOSSE 不同的是，STC 在训练模板、即计算 hsc(x−z)hsc(x−z)时只需考虑第一帧。而在跟踪过程中，hsc(x−z)hsc(x−z)的更新方式如同 MOSSE，这里不再叙述。另外论文中还给出了目标框大小更新的方法，其基本思路可以这样理解：看到公式(5)m(x)=∑z∈Ωc(x★)hsc(x−z)I(z)ωσ(z−x★)m(x)=∑z∈Ωc(x★)hsc(x−z)I(z)ωσ(z−x★)，其中 ωσ(z−x★)ωσ(z−x★)不就是高斯形状的权重嘛，稍微不恰当的说，就是用个圆圈把目标包住嘛，圈内的权重高，圈外的相反，那么假如目标的 size 变大了，我们就把这个圈的范围扩大就好了，而扩大或者缩小就靠调整 σσ 的值就 ok 了。具体推导过程如下：



假设从 tt 到 t+1t+1 帧，目标的大小乘以了一个 ss 倍，也即相当于坐标系的刻度乘以了 ss 倍，为方便起见，我们设 (u,v)=(sx,sy)(u,v)=(sx,sy)，然后，不失一般性的，假设目标在第 tt 帧的坐标为(0,0)，则有

$$\begin{aligned} c_t(0,0) &= \iint_{x,y} h_t^{sc}(x,y) I_t(x,y) \omega_{\sigma_t}(x,y) dxdy \\ &= \frac{1}{s^2} \iint_{u,v} h_t^{sc}(u/s, v/s) I_t(u/s, v/s) \omega_{\sigma_t}(u/s, v/s) dudv \end{aligned} \tag{8}$$

由 ωσ(x,y)=αe−x2+y2σ2,ωσ(x/s,y/s)=αe−x2+y2(sσ)2ωσ(x,y)=αe−x2+y2σ2,ωσ(x/s,y/s)=αe−x2+y2(sσ)2 有 ωσ(x/s,y/s)=ωsσ(x,y)ωσ(x/s,y/s)=ωsσ(x,y)，所以(8)式继续推导为：

$$c_t(0,0) = \frac{1}{s^2} \iint_{u,v} h_t^{sc}(u/s, v/s) I_t(u/s, v/s) \omega_{s\sigma_t}(u,v) dudv \tag{9}$$

然后，从 tt 变到 t+1t+1 帧，我们把变化后的坐标对应起来，因此有 hsct(u/s,v/s)≈hsct+1(u,v)htsc(u/s,v/s)≈ht+1sc(u,v)和 It(u/s,v/s)≈It+1(u,v)It(u/s,v/s)≈It+1(u,v)，所以式(9)继续变为

$$\begin{aligned} c_t(0,0) &\approx \frac{1}{s^2} \iint h_{t+1}^{sc}(u,v) I_{t+1}(u,v) \omega_{s\sigma_t}(u,v) dudv \\ &= \frac{1}{s^2} \iint_{u,v} h_{t+1}^{sc}(u,v) I_{t+1}(u,v) \omega_{s\sigma_t}(u,v) dudv \end{aligned} \tag{10}$$

假设从 tt 到 t+1t+1 帧是缩小的，因此跟缩放示意图一样，我们将(10)的积分看成两部分组合成的：一是红框部分(t+1t+1 帧的上下文框大小)，二是蓝框(tt 帧的上下文框大小)减去红框的部分，用公式表达就是：

$$\begin{aligned} c_t(0,0) &= \frac{1}{s^2} \iint_{u,v} h_{t+1}^{sc}(u,v) I_{t+1}(u,v) \omega_{s\sigma_t}(u,v) dudv \\ &= \frac{1}{s^2} \Big(\iint_{u,v \in \Omega_{sx,sy}} h_{t+1}^{sc}(u,v) I_{t+1}(u,v) \omega_{s\sigma_t}(u,v) dudv + \iint_{u,v \in \Omega_{x,y} - \Omega_{sx,sy}} h_{t+1}^{sc}(u,v) I_{t+1}(u,v) \omega_{s\sigma_t}(u,v) dudv \Big) \end{aligned}$$

(11)

又因为 ωω 的高斯形状的关系，上式右项那一部分的权值都很小，因此整个右项都可视为 0，同时将 sσtsσt 视为 σt+1σt+1，所以上式的左项就近似成了 ct+1(0,0)ct+1(0,0)：

$$c_t(0,0) \approx \frac{1}{s^2} \iint_{u,v \in \Omega_{sx,sy}} h_{t+1}^{sc}(u,v) I_{t+1}(u,v) \omega_{\sigma_{t+1}}(u,v) du dv = c_{t+1}(0,0) \tag{12}$$

因此就有

$$s = \sqrt{\frac{c_{t+1}(0,0)}{c_t(0,0)}} \tag{13}$$

剩下的就是一些技巧了，比如用滑动窗口取 ss 的平均之类的，具体可以看作者的原文。这篇文章大概就到这里了。总结一下，其中比较吸引笔者的其中的概率论支撑和后面的窗口大小的变化部分，至于环境上下文部分的话，换用其它特征应该可以作进一步扩展以提高算法的鲁棒性。作者主页上有源代码，有兴趣的可以下载来跑跑看，运行时留意下像 woman 这类视频吧~