Published by :
http://www.ijert.org

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 7 Issue 02, February-2018

Video Inpainting Technique using Non Local Patch Based Method

E. Cinthuriya
ME (Communication Systems) Student,
Department of ECE
E.G.S. Pillay Engineering College
Nagapattinam

M. Nuthal Srinivasan M.E.,
Assistant Professor
Department of ECE
E.G.S. Pillay Engineering College
Nagapattinam..

***Abstract* — Video inpainting is a growing area of research and it's the method of eliminating specific unwanted areas or repairing the missing or damaged area in a video sequence. It is a unique method for filling holes or missed areas in a video. This can be applicable to both static or slow moving camera videos. This technique is useful for object tracking, objects removal and loss concealment purposes. There has been several video inpainting methods and algorithms proposed so far. In this project, I propose an automatic video inpainting algorithm which is based on the non local patch-based technique. The algorithm is able to deal with a variety of challenging situations, such as the correct reconstruction of dynamic textures, moving objects and moving background. Aim of this work is to inpaint a video with less execution time and also achieves good quality results on high definition videos and complex scene videos.**

Key words* — *Patch Based Method, Inpainting, Dynamic Texture.

I. INTRODUCTION

Video Inpainting refers to a field of Computer Vision that aims to remove objects or restore missing or tainted regions present in a video sequence by utilizing *spatial* and *temporal* information from nneighbouring scenes.

The overriding objective is to generate an inpainted area that is merged seamlessly into the video so that visual coherence is maintained throughout and no distortion in the affected area is observable to the human eye when the video is played as a sequence. It is important at this stage to distinguish video and image inpainting from the related field of texture synthesis. The main difference between these approaches is in the size and characteristics of the region to be corrected. For texture synthesis the region can be much larger with the main focus being the filling in of two-dimensional repeating patterns that have some associated textures. In contrast, inpainting algorithms concentrate on the filling in of much smaller regions that are characterised by linear structures such as lines and object contours. There has been some recent work by Criminisi et al. to bridge this gap and to provide a single algorithm to handle both of these cases given both the presence of textures and structures in images. Despite still being a relatively young field compared to the related area of image inpainting, some rather encouraging results have already been obtained by the research community.

A. Reason For Not Using Methods From The Field Of Image Inpainting

It is important to clarify the reason as to why it is simply not possible to extend methods put forward for image inpainting (that is, the completion of damaged regions in single images using information from the surrounding areas) to the related field of video inpainting. After all a video can just be thought of as a sequence of still images. The key point to note is that image inpainting techniques only consider *spatial* information and completely neglect the significant *temporal* component present in video sequences.

Only considering the spatial component leads to quite severe image artefacts being produced in the resulting inpainted video. This is because the essential assumption of image inpainting, that edges should be interpolated in some smooth way, is not valid when we consider the dimension of time. Non-smooth temporal changes can arise, for instance, due to a particular pixel containing background in one frame, and fore ground in a following frame. It may well be the case that when the inpainted video frames are taken separately no artefacts are visible, but only when played in a sequence do the artefacts (for example, a phantom object drifting steadily through the video) manifest themselves to the human eye.

II. VIDEO INPAINTING APPROACHES

Although, the image inpainting problem has been extensively studied in the literature, there is much less works for video content. This can be explained by the fact that when dealing with inpainting problem in video, not only the spatial coherence should be taken into account but also the temporal continuity among video frames needs to be ensured which introduces high time complexity. Thus, simply applying image inpainting approaches is not sufficient to provide pleasing results. Nevertheless, video inpainting methods are often the extensions of image completion algorithms by using additional constraints that aim to provide both space and temporal consistency. The temporal information is either considered by using a segmentation of video objects (e.g. tracking) or a global coherency in space-time patches. Besides, the filling of the hole can be performed based on greedy or globally optimized method. In the following subsections we describe the widely cited approaches in the literature broadly classified into local and global optimizing methods.

B. Globally optimized approaches

This category solves the inpainting problem by using a single energy function whose minima provides globally consistent results. Similarly to the image, each pixel or patch in the hole is associated to the optimal unoccluded values

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 7 Issue 02, February-2018

based on the similarity between 2D or 3D patches in videos data. A well-known method of this category has been proposed by Wexler et al. This approach solves the video inpainting problem of static camera videos based on the optimization of a well-defined energy function. In a first step, space-time source patches are sampled to provide a database of patches statistics in the video. Then, the completion is performed by optimizing the cost function expressing the local consistencies by using a weight of the global completion quality provided by each possible pixel value.

Each missing pixel is then assigned the most likely color among the patches database that optimizes the global coherence function. This approach can handle both structured and textured scenes and provide high inpainting results without any segmentation. However, the processing time is high even for low resolutions videos. The complexity of the search for the best matching patches has been reduced by using an extension of the patch matching algorithm in to the video.

In Shen et al. proposed also a global optimizing inpainting approach with low computational complexity by tracking every pixel in the occluded area through the video. The searching window for each missing pixel is then reduced from 3D to a 2D manifold provided by the tracking step. Unfortunately, this method can handle only translational or periodic objects motions.

C. Locally optimized approaches

The locally or greedy optimized approaches are usually based on video segmentation into moving foreground objects and background to be inpainted using separate algorithms. This idea has been introduced by Patwardhan et al. extending examplar-based image inpainting approach in to solve the video completion problem. The inpainting of the hole is performed with a pre-processing stage followed by three inpainting steps.

In the pre-processing step the frames are segmented into moving objects (foreground) and static background. Then, the inpainting algorithm starts by filling in the missing regions corresponding to moving objects. The remaining holes which mainly correspond to the static background are first recovered by exploiting similarity between frames (temporal redundancy). Finally, the spatial inpainting in is used for remaining holes. In each step, the priority of every patch located on the front line is computed based on the local amount of undamaged pixels and on the motion direction. Proceeding by highest priority, the method copies those patches that best match with the target patch with respect to the known pixels.

The similarity between patches is here the distance between the patches textures and motion vectors. This method has been later extended to handle translational camera motions in and more general camera motions .

Object-based hole completion proposed by Venkatesh et al. is based on tracking the occluded object and building a database of segmented frames where the occluded object is fully visible. The missing regions are then completed by aligning frames in the database to the partially or fully occluded frames in the hole. The contour of the occluded object is then introduced by Ling et al. to retrieve the object frames from the database. Query postures are synthesized based on local segments of the object to find database frames. Similarly, Jia et al. inpaint the missing holes of the occluded objects by aligning the object's visible trajectory.

Other approaches based on transfer of motion fields into the hole either by gradually propagating motion vectors in the border area or by using motion similarities between patches.

These methods are likely to suffer from smoothing artificial after few frames making the approach not well suited for completion over a high number of frames.

III. PREVIOUS WORKS

Video inpainting is a particularly challenging problem. Difficulties arise due to the possibilities of camera motion, the requirement to handle both static and dynamic regions, as well as the pose and scale variation, lighting and shadows, background clutter, deformation and occlusions that are commonly present in a video. The problem is usually made tractable by imposing some limiting constraints and solving for a less complex subclass of the problem. Common constraints include a static camera, the handling of small ranges of motions only, and a limited size of region to be repaired(commonly this is quite small).The literature is replete with a varied array of video inpainting proposals each advocating different approaches to the problem, some of which we will briefly mention here under. Bertalmio et al.I investigated the extent to which ideas from computational fluid dynamics could be applied to image and video inpainting. Here the authors only consider spatial information in a video and perform the inpainting on a frame by frame basis leading to limited applicability of the algorithm. In contrast, Wexler et al. take account of the spatial and temporal dimensions in their space-time video completion approach which enables the model to handle video sequences of complex dynamic scenes. Here the authors attempt to solve the inpainting problem by sampling a set of spatial-temporal patches (a set of pixels at frame t) from other frames to fill in the missing data. Global consistency is enforced for all patches surrounding the missing data so as to ensure coherence of all surrounding space time patches. This avoids artefacts such as multiple recovery of the same background object and the production of inconsistent object trajectories. This method provides decent results, however it suffers from a high computational load and requires a long video sequence of similar scenes to increase the probability of correct matches. Patwardhan et al. suggest a rather simpler method for inpainting stationary background and moving foreground in videos. To inpaint the stationary background irrelatively simple spatio-temporal priority scheme is employed where undamaged pixels are copied from frames temporally close to the damaged frame, followed by a spatial filling in step which replaces the damaged region with a best matching patch so as to maintain a consistent background throughout the sequence. Jia et al. propose an inpainting method that aims to tackle the challenging issue of inpainting under the presence of severe occlusion. The authors segment a video into a moving object layer and a static background layer. The techniques of layer segmentation and homographs blending are used to recover static background pixels. Moving foreground pixels are repaired by sampling motion data followed by the

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 7 Issue 02, February-2018

application of 3D tensor voting so as to maintain temporal coherence and motion periodicity. Missing dynamic foreground pixels are then inferred by spatial and temporal alignment of the sampled motion data.

This work is significant more complex than other proposals in the literature, but this added complexity enables the algorithm to handle a range of camera motions, including zoom-in, zoom-out, rotation about a fixed point, and panning. It is quite common for most of the approaches in the literature to require the user to specify the damaged region to be repaired. However in [6], the authors manually show how it is possible to identify regions to be repaired in an automatic fashion using motion cues. This is certainly an important step for the widespread adoption of video inpainting technologies amongst the commercial and private communities. In this article we will discuss the video inpainting proposal put forward by Patwardhan et al. which describes a simple, fundamental approach to the problem making it ideal for the purposes of introducing and illustrating the core concepts in the field. The reader is encouraged to consult the further reading references at the end of this article for pointers to other papers describing the alternative methods of video inpainting.

IV. THE PROPOSED WORK.

The ultimate goal of our work is to produce an automatic and generic video inpainting algorithm which can deal with varied situations. First of all, we tackle the problem of long execution times, which is a significant obstacle for research in video inpainting. This is due to the heavy computational load of the search for the nearest neighbours of video patches. This problem by proposing an extension of the Patch Match algorithmm to the spatio-temporal case, which greatly accelerates this nearest neighbour search. Dealing with this problem is very important, as it is very difficult to test and experiment with the algorithm without reasonable execution times Secondly,we look in detail at inpainting videos which contain dynamic video textures. Indeed, the reconstruction of textures and fine details is one of the main limitations of video inpainting methods. To deal with this, we propose a modification to the patch distance in order to identify these textured areas. We also create a multi-resolution texture feature pyramid to aid the correct reconstruction of textures at all resolutions.

A. Non Local Patch Based Approach

Proposed video inpainting algorithm takes a non-local patch-based approach. At the heart of such algorithms lies a global patch-based functional which is to be optimised.

We optimise this with an iterative algorithm, as is commonly found in algorithms using such a formulation of the problem finding in frames. The central machinery of the algorithm is based on the alternation of two core steps: a search for the nearest neighbours of patches which contain occluded pixels, and a reconstruction step based on the aggregation of the information provided by the nearest neighbours.This iterative algorithm is embedded in a multi-resolution pyramid scheme. The multi-resolution scheme is vital for the correct reconstruction of structures and moving objects in large occlusions.

In our work, we also propose an additional multi-resolution pyramid which reflects textural attributes of the video, so as to correctly reconstruct video textures. This second pyramid is reconstructed in parallel to that of the video colour information. It is important to point out that both pyramids contribute to the nearest neighbour search, and therefore jointly determine the reconstruction of the colour and texture attributes at each iteration.

B. Description of algorithm

As discussed above, the core of our algorithm is the minimization of a patch-based non-local functional. Before describing the algorithm in detail, first of all we need to set out some notation.

C. Notation

The image support is denoted with Ω, and this support is separated into two regions: H the occlusion(H for “hole”) and D the unoccluded region (D for “data set”), such that H ∩D = ∅ and H ∪D = Ω.A position in the support may be denoted with p = (x, y). The image content itself is denoted with u : Ω → R3.Another very important notion is that of a patch. We first define the patch neighbourhood of a pixel p as a set of positions in Ω, denoted with Np, of cardinality N. The patch itself is denoted with

$$Wp= (u(x1), \cdots , u(xN)). \quad (1)$$

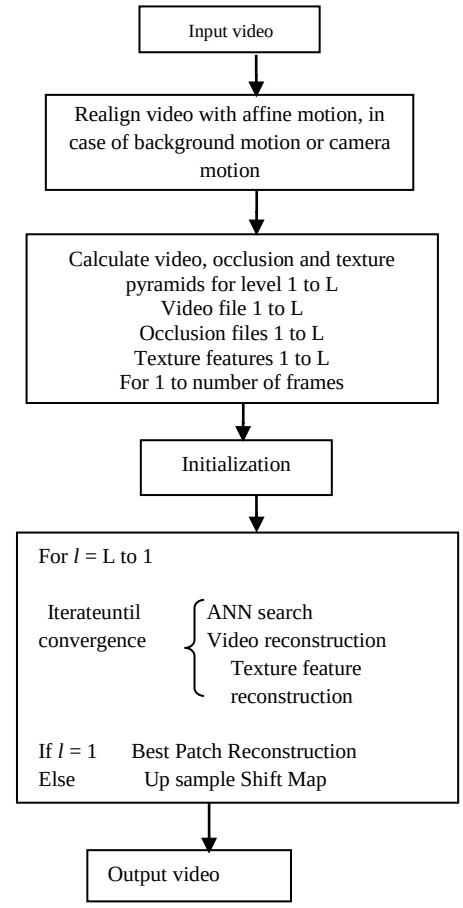

fig. 4.1. flowchart of proposed algorithm

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 7 Issue 02, February-2018

A connected notion which we need to define is that of a nearest neighbour(NN for short) of a patch Wp. The NN of Wp is a patch Wq in another set of patches which minimisesa certain patch distance d(Wp, Wq). We shall choose NNs from the set of patches which contain nooccluded pixels. For this, define D˜ the set of positions such that ∀p ∈ D˜, Np ⊂ D. Therefore, theNN of Wp is Wq, with

$$E(u,ø)=\sum_{p\in H} d^2(W^u_p, W^u_{p+ø(p)}) \quad (1)$$

q = argmin
k∈D˜
d2(Wp, Wk).

Finally, we define the shift map φ : Ω → N2 as a vector field which shows where the NN of a patch is located. That is, the NN of Wp is Wp+φ(p).

D. Variational framework

The functional which we wish to minimise takes the following form.

Intuitively speaking, this highly non-convex functional specifies that for a "good" solution, each patch inside H should be as similar as possible (in terms of the patch distance d2(·, ·)) to its nearest neighbour in the unoccluded region D. We use the following distance:

$$d^2(W^u_p, W^u_{p+ø(p)})=(1/N)\sum_{q\in Np} \| u(q)-u(q+ø(p)) \|^2_2. \quad (2)$$

Following the heuristic proposed in wel., space time video, a solution is obtained using an iterated alternating minimisation, as well as a multi-scale framework. At each scale, we minimise firstly with respect to the shift map φ, then with respect to the image content u. These two steps correspond, respectively, to the following operations:

- Nearest neighbour search.
- Image reconstruction.

The nearest neighbour search can be quite expensive, especially if the whole image is used as the search space. In Section 4.1, we specify the manner in which the NN search is carried out, and how the computational issue is addressed. The reconstruction process is described in Section 4.6. An improvement of this basic process is proposed in Section 4.7in order to improve the reconstruction of textures. Section 4.8deals with the initialisation step, and Section 4.9deals with the multi-scale framework of the algorithm, both points being especially important given the non-convex nature of the problem. Now that we have given a broad outline of our approach, we provide the specific algorithmic details necessary for the implementation of each component of our algorithm.

D. Approximate Nearest Neighbour Search

As mentioned previously, the NN search can be quite computationally expensive. In such situations,it is common to use approximate nearest neighbours instead of exact nearest neighbours. In this implementation, we use the Patch Match algorithm to search for nearest neighbours. This algorithm was introduced by Barnes et al. [3] and relies on the idea that good shift maps tend to be piece wise constant. In other words, a relative shift which leads to a good NN for a patch Wp has a good chance of leading to a good NN for the patches situated around Wp.

The algorithm consists of three steps:

- Random initialisation.
- Propagation.
- Random search.

The first step is carried out once at the beginning of the algorithm. Each occluded pixel is randomly assigned a candidate NN in the region D˜. The next two steps are carried out on each patch, and each patch is visited in lexicographical order. This is carried out a predefined number of times. The propagation step attempts to spread good shifts throughout the vector field φ. Finally, the random search step looks for better NNs randomly in an increasingly small window around the current NN. Since Patch Match itself is not a novelty of our algorithm, we refer the reader to the original publication for further details.In practice, we only need the initialisation step during the filling in of the occlusion (at the beginning of our algorithm). After this, we may consider that we have a good initialisation of φ, which is simply the φ of the previous iteration. Let us emphasize that Patch Match has already been used as a way to accelerate the search for ANN in a commercial implementation of the algorithm from[19], in the content aware tool from the software Photoshop.

The pseudo-code for the Patch Match algorithm may be seen in Algorithm 1.

Data: Current inpainting configuration u (height: m, width: n), φ, H˜
Parameters: rmax (max(m, n)), ρ (0.5)
Result: ANN shift map φ
for k = 1 to k(max) do
for p = p1to p|H| ˜ (pixels in H˜ lexicographically ordered) do
If k is even then (propagation on even iteration)

a=p-(1,0),b=p-(0,1);

q = arg minr∈{p,a,b} d(Wpu, Wpu+φ(r));
if p + φ(q) ∈ D˜ then φ(p) ← φ(q);
else /* Propagation on odd iteration */

a = p + (1, 0), b = p + (0, 1);
q = arg minr∈{p,a,b} d(Wpu, Wpu+φ(r));
if p + φ(q) ∈ D˜ then φ(p) ← φ(q);
end
zmax ← ⌈- log(max(log(ρm,n)))⌉;
for z = 1 to zmax do
q = p + φ(p) + ⌊rmaxρzRandUniform([-1, 1]2) ⌋;
if d(Wpu, Wpu+φ(q)) < d(Wpu, Wpu+φ(p)) and p + φ(q) ∈ D˜ then
φ(p) ← φ(q);
end
end
end

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 7 Issue 02, February-2018

E. Reconstructing The Image

The second step in the iterative algorithm is the minimisation of the functional in Equation (1) with respect to u. Intuitively speaking, this is the process of assigning a colour value to each position in the occluded area, or as we refer to it here, the reconstruction process. The minimisation of (1) with respect to u should lead to each pixel being reconstructed with an un weighted mean of several colour values in D. However, we shall use a weighted mean scheme, initially proposed by Wexler et al. Experimentally, we have observed that this improves the convergence of the algorithm.

- Non-local patch-based image inpainting
- Given a shift map, each pixel is reconstructed as:

$$u(p)=(\sum_{q\in Np} s^p_q u(p+ø(q)))/(\sum_{q\in Np} s^p_q) \quad (3)$$

where s^p_q is the weight assigned to the pixel value for p indicated by the ANN of

$$s^p_q=\exp((-d^2(w_q,w_q+ø(q)))/(2\sigma^2) \quad (4)$$

Informally, it can be seen that each pixel p is reconstructed using the "versions" of p indicated by the ANNs of all the patches which contain p. This operation is carried out for all pixels p ∈ D.An inevitable problem of such a reconstruction scheme is blurred results, especially in textured areas. This was noticed by Wexler et al. in [19], who proposed a method to avoid this effect which used the mean shift algorithm. The idea is to gradually decide on the best value for each pixel by clustering the colour values and only using pixels which correspond to the dominant cluster for reconstruction.

After experimentation, we found that such a sophisticated approach was unnecessarily complex, and a very similar result could be obtained simply by choosing the "best" pixel at the end of the algorithm (at the finest pyramid level) when it has converged. Therefore the final reconstruction step is the following by equation.

$$U(p)=u(p+ø(q^*)),\text{with } q^* = argmin\ d^2(W_q,W_{q+ø(q)}) \quad (5)$$

F .Inpainting With Textures

Even though patch-based methods were initially introduced for texture synthesis, patch-based reconstruction of textured areas is often problematic when dealing with composite scenes made of different regions.. More precisely, when several regions with a similar average colour but differing texture contents exist, three main problems can arise with multi-scale and iterative algorithms:

- .Ambiguities in patch comparisons at coarse pyramid levels;
- Failure of the regular ℓ2 patch distance to correctly compare textures
- The weighted mean reconstruction tends to smooth textures during thealgorithm, which in turn leads to smooth patches being chosen as NNs.

In order to address these problems, we introduce texture features into the patch distance. The goal of these features is to correctly match textured patches with similarly textured patches. After experimenting with several different features, we found that the following features, inspired by the work of Liu and Caselles , identified textures in a satisfactory manner for our purposes:

$$T_x(p)= 1/(\text{card}(v).\sum_{q\in v} \left| I_x(p) \right|$$

$$T_y(p)= 1/(\text{card}(v).\sum_{q\in v} \left| I_y(p) \right| \quad (6)$$

where v is a neighbourhood centred on p, and Ix and Iy are the derivatives of the image in the x and y directions. In the case of white noise patches with a Gaussian distribution, these attributes tend to the variance of the noise (up to a multiplicative factor) as the size of v increases.

Therefore, we consider our texture to be well-represented by the variance of its pixel values. This is obviously quite a simple representation of a texture, but we found that it distinguishes well between textured and non textured areas with the same average colour.
Thus, the squared patch distance is now redefined as:

$$d^2(W_p,W_q)=(1/N)\sum_{(r\in Np)}(\left\| u(r)-u(r-p+q)) \right\|^2_2+\lambda \left\| T(r)-T(r-p+q) \right\|^2_2). \quad (7)$$

It is important to note that we calculate these features at the finest pyramid level for the unoccluded pixels p ∈ D˜, and propagate them to coarser levels by nearest neighbour sub sampling. This is necessary since the textures are not present at coarser levels, the image having been smoothed and sub sampled. This solves the problem of correctly matching textured patches, but due to the patch averaging process in Equation 4, we still have a tendency to smooth the result. This gradually leads to inpainting with the wrong texture, in spite of the process detailed in Section 3.2. To deal with this, we create a separate image with the texture features, and this image is inpainted in parallel to the colour image.

$$T(p)=(\sum_{q\in Np} s^p_q T(p+ø(q)))/(\sum_{q\in Np} s^p_q) \quad ,\forall_p \in H. (8)$$

G. Initialisation

All iterative methods which attempt to optimise an objective function require an initial solution, and this initialisation turns out to be very important in our case (which is highly non-convex, as mmentioned previously). It is also an issue which is seldom discussed in many works on the subject.

We propose an "onion-peel" approach which in paints the occlusion one layer at a time, eroding the occlusion progressively. The layers are one-pixel thick, and located at the border of the occlusion. Each pixel in a layer is processed using the information of the inpainting solution of the previous layer, and thus there is no processing order for pixels within a single layer (they are processed "in parallel"). For such an approach, we obviously need to use a partial patch comparison, since the content of the patches will not be completely known. Let H′ ⊂ H be the current occlusion,

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 7 Issue 02, February-2018

and $\partial H' \subset H'$ the current layer to inpaint. We now define the partial patch neighbourhood :

$$N'_p=\{q \in N_p, q \neq H'\} \quad (9)$$

The distance between two partially known patches Wp and Wp+φ(p) is defined as:

$$d^2(W_p,W_q)=(1/N'_p)\sum_{(q \in N'_p)}(\| u(q)-u(q+ø(p)) \|^2_2+\lambda \| T(q)-T(q+ø(p) \|^2_2) \quad . (10)$$

Now that we are able to compare two partially known patches, we wish to reconstruct the information of a given pixel p in the current occlusion layer $\partial H'$. For this, we need to choose which patches containing p we will use for the reconstruction. Indeed, the patches will contain more or less known pixels, and thus be more or less reliable.

We propose to use only those patches which are centred at known pixels. Thus, we have:

$$u_p=(\sum_{q \in N'_p} s^p_q u(p+ø(q)))/ (\sum_{q \in N_p} s^p_q) \quad (11)$$

The texture features are reconstructed in the same fashion. In this case, as in many of our other experiment, It is clear that the use of an onion peel approach provides the best initialisation when compared with other common approaches.

H. Multiscale scheme

The use of a multiscale scheme is of crucial importance in avoiding local minima of the energy functional (1), and in converging to more satisfactory solutions. Therefore, care must be taken in Algorithm 2: Complete proposed inpainting algorithm.

Data: Input image u, occlusion H ;
Result: Inpainted image
{uℓ}L ℓ=1 ←ImagePyramid(u);
{T ℓ}L ℓ=1 ←TextureFeaturePyramid(u); // Eq.7
{Hℓ}L ℓ=1 ←OcclusionPyramid(H);
φL ←Random;
(uL, T L, φL) ←Initialisation(uL, T L, φL, HL); // Eq.
for ℓ = L to 1 do
k = 0, e = 1;
while e >0.1 and k <20 do
v = uℓ;
φℓ ←ANNsearch(uℓ, T ℓ, φℓ, Hℓ) ; // Alg.1
uℓ ←Reconstruction(uℓ, φℓ, Hℓ) ; // Eq.3
T ℓ ←Reconstruction(T ℓ, φℓ, Hℓ);
e = 1
3Hℓ|kuℓ Hℓ - vHℓk2;
k ← k + 1;
end
If ℓ = 1 then
u ←FinalReconstruction(u1, φ1, H) ; // Eq.5
else
φℓ-1 ←UpSample(φℓ, 2) ; // Sec.3.5
uℓ-1 ←Reconstruction(uℓ-1, φℓ-1, Hℓ-1);
T ℓ-1 ←Reconstruction(T ℓ-1, φℓ-1, Hℓ-1);
end
end

Non-local patch-based image inpainting its implementation. The sub-sampling factor is 1 2, and we use a Gaussian 3 × 3 filter with a standard deviation of 1.5 pixels for smoothing. Two further details are of particular importance: the manner in which an inpainting solution is passed from a coarser pyramid level to a finer one, and the number of pyramid levels to use. In our implementation, we up sample the shift map φ rather than the inpainting solution itself. We do this to avoid the inevitable smoothing which would happen if we took the second option. Indeed, the correct textures are not present in the coarser level (as we mentioned in Section 4.5), and therefore using this as an initialisation could encourage the algorithm to use smoother areas for inpainting.

The shift map is up sampled using a simple nearest neighbours interpolation, with non integer coordinates resulting from this interpolation being rounded down to the nearest integer. Then, the initial solution is produced by reconstructing the image at the new resolution with Equation (3).The same process is applied to the texture features.
The next choice which must be made is the number of pyramid levels. This subject has been given very little thought in the inpainting literature, and generally the number of levels is decided by fixing a minimum image resolution and sub sampling until that resolution is attained.

In reality, this heuristic is not a very good way to ensure that good minima are produced. For example, if the present resolution is already attained in the original image, and the occlusion size is large, then structures will not be correctly continued into the occlusion.

A better criterion is the relationship between the patch size and the occlusion size. Roughly speaking, the patch size needs to “cover” a certain amount of the occlusion for occlusion size should be a little more than twice the patch size for maintaining structures. In practice, the occlusion “size” is not necessarily trivial to calculate.

To determine this, we iteratively erode the occlusion with a square structuring element of width three pixels until it disappears. Let No be the number of times that the occlusion needs to be eroded for it to disappear. We define the number of pyramid levels as:

$$L=(\log(2N_0/N))/\log(2) \quad (12)$$

This method was used for all of our experiments.This concludes the algorithmic description of the proposed method. The pseudo-code for the complete algorithm may be seen in Algorithm 2.

I. Algorithm parameters

The main tunable parameter of our algorithm is the patch size. Unless specified otherwise, we usea default patch size of 7 × 7, which is adequate for most images we used (from around 512 × 512 to800 × 800 pixels).The parameters

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 7 Issue 02, February-2018

of the Patch Match Algorithm are considered to be fixed. We set the random search reduction window to be ρ = 12. We have used ten iterations of propagation/random search in Patch Match, although this can be reduced if quicker results are required.We use the procedure described in Section 4.5 to determine the number of pyramid levels automatically, although this can be set manually if required. We set the weight λ associated with the texture features to 50. Finally, we identify the convergence of the algorithm at a pyramid level by determining the average absolute pixel value difference for each colour channel between the current solution and the previous one. If this value drops below 0.1 or the number of iterations exceeds ten, we stop iterating at the current level.

V. EXPERIMENTAL RESULTS

We now turn to another common case of video inpainting, that of mobile backgrounds. This is the case, for example, when hand-held cameras are used to capture the input video.

Object Removal in a Video:

Beach Umbrella video (264 X 68)

In this video, object removal is done by using video inpainting approach. Removal of beach umbrella is achieved by converting original video into number of frames. For this we get totally 98 frames.

The Fig.5.1 and 5.2 shows original and inpainted frame in this video. For example 30th frame is taken out of 100 frames.

fig.5.1. beach umbrella frame number 30

fig.5. 2 beach umbrella frame number 30

VI. CONCLUSIONS AND FUTURE WORK

In this paper, we have proposed a non-local patch-based approach to video inpainting which produces good quality results in a wide range of situations, and on high definition videos, in a completely automatic manner. Our extension of the Patch Match ANN search scheme to the spatio-temporal case reduces the time complexity of the algorithm, so that high definition videos can be processed. We have also introduced a texture feature pyramid which ensures that dynamic video textures are correctly inpainted.

The resulting algorithm performs well in a variety of situations, and does not require any manual input or segmentation. In particular, the specific problem of inpainting textures in videos has been addressed, leading much more realistic results than other algorithms. Video inpainting has yet not been extensively used, in a large part due to prohibitive execution times and/or necessary manual input. We have directly addressed this problem in the present work. We hope that this algorithm will make video inpainting more accessible to a wider community, and help it to become a more common tool in various other domains ,such as video post-production, restoration and personal video enhancement.

In future work we deal with the problem of moving background, using a robust affine estimation of the dominant motion in the video. This turns out to be a crucial point to resolve, even in the case of relatively small motions, such as the shaking of a hand-held camera.

REFERENCES

[1] Alasdair Newson, Andr´ es Almansa, Matthieu Fradet, Yann Gousseau, and Patrick P´ erez,Towards fast, generic video inpainting, in Eur. Conf. Visual Media Production (CVMP), 2013

[2] Acoustic, Speech and Signal Processing (ICASSP), 2017 IEEE International Conference on Spatio-temporal binary video inpainting via threshold dynamics 29 June 2017.

[3] Anil K. Jain, , Fundamentals of Digital Image Processing', Pearson Education,Inc., 2002.

[4] Non-local patch-based image inpaintingAlasdair Newson1, Andr´es Almansa2, Yann Gousseau2, Patrick P´erez3Universit´e Paris Descartest PArdeshir Goshtasby, “2D and 3D Image regireprint november 11, 2015

[5] Space-Time Completion of VideoYonatan Wexler, Member, IEEE Computer Society,Eli Shechtman, Student Member, IEEE Computer Society, andMichal Irani, Member, IEEE Computer Society

[6] Mounira Ebdelli, Olivier Le Meur, Christine Guillemot. Video inpainting with short-termwindows: application to object removal and error concealment. IEEE Transaction on ImageProcessing, IEEE, 2015, 24 (10), pp.3034-47.

[7] C. Guillemot and O. Le Meur, “Image inpainting: Overview and recent advances,” *IEEE Signal Process. Magazine*, Jan. 2014.

[8] Ravi, Siddarth, et al. "Image in-painting techniques-A survey and analysis."*Innovations in Information Technology (IIT), 2013 9th International Conference on*. IEEE, 2013.

[9] Stration for Medical, Remote Sensing and Industrial Applications”,John Wiley and Sons,2005.

[10] C.-H. Ling, C.-W. Lin, C.-W. Su, Y.-S. Chen, and H.-Y. M. Liao, “Virtual contour-guided video object inpainting using posture mapping and retrieval,” *IEEE Transactions on Multimedia*, vol. 13, no. 2, pp. 292–302, Apr.2011.

[11] Chih-Hung Ling,Yu-Ming Liang,Chia-Wen Lin,Yong-Sheng Chen,Hong-Yuan Mark Liao,” Human object inpainting using manifoldlearning-based posture sequence estimation”, *IEEE*

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 7 Issue 02, February-2018

Transactions on Image Processing, vol.20,no.11,pp. 3124 - 3135 , 2011.

[12] J. Jia, Y. Tai, T. Wu, and C. Tang, "Video repairing under variable illumination using cyclic motions," *IEEE Intelligence*, vol. 28, no. 5, pp. 832–883, May 2006.

[13] J. Jia, Y.-W. Tai, T.-P. Wu, and C.-K. Tang, "Video inpainting using posture mapping," *IEEE Transactions on Pattern Analysis and. Machine Intelligence,* vol. 28, no. 5, pp. 832–839, May 2006.

[14] K.A. Patwardhan, G. Sapiro, and M. Bertalmío, "Video inpainting under constrained camera motion," *IEEE Transactions on Image Processing,* vol.16, no. 2, pp. 545–553, February 2007.

[15] Mounira Ebdelli, Olivier Le Meur, and Christine Guillemot "Video Inpainting With Short-Term Windows: Application to Object Removal and Error Concealment" *IEEE transactions on image processing*, vol. 24, no. 10, october 2015

[16] S.-C. S. Cheung, J. Zhao, and M. V. Venkatesh, "Efficient object-based video inpainting," *Proceedings of IEEE Conference on Image Processing*, pp. 705–708,Oct. 2006

[17] T. K. Shih, N. C. Tang, and J.-N. Hwang, "Exemplar-based video inpainting without ghost shadow artifacts by maintaining temporal continuity,"*IEEE Transactions on Circuits and Systems for Video Technology*, vol. 19, no. 3, pp.347–360, Mar. 2009.

[18] Y. Shen, F. Lu, X. Cao, and H. Foroosh, "Video completion for perspective camera under constrained motion," *Proceedings of IEEE Conference on Pattern Recognition*, pp. 63–66, August 2006.

[19] Y. Wexler, E. Shechtman, and M. Irani, "Space-time completion of video," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 29, no. 3, pp. 463– 476, March 2007.

[20] Y.-T. Jia, S.-M. Hu, and R. R. Martin, "Video completion using tracking and fragment merging," *Proceedings of Pacific Graphics*, vol.21, no. 8–10, pp. 601–610, 2005.